UNIVERSIDAD DE LA RIOJA

TRABAJO FIN DE ESTUDIOS

Título

Diseño, desarrollo y construcción de sistema de control para servoválvula

Autor/es

MIGUEL DE GREGORIO SANTAMARINA

Director/es

JAVIER RICO AZAGRA

Facultad

Escuela Técnica Superior de Ingeniería Industrial

Titulación

Grado en Ingeniería Electrónica Industrial y Automática

Departamento

INGENIERÍA ELÉCTRICA

Curso académico

2017-18

publicaciones.unirioja.es
E-mail: publicaciones@unirioja.es

UNIVERSIDAD DE LA RIOJA

ESCUELA TÉCNICA SUPERIOR DE INGENIERÍA INDUSTRIAL

TRABAJO DE FIN DE GRADO

TITULACIÓN: Grado en
Ingeniería Electrónica Industrial y Automática

CURSO: 2017/2018 **CONVOCATORIA: JULIO**

TÍTULO:

DISEÑO, DESARROLLO Y CONSTRUCCIÓN DE SISTEMA DE CONTROL PARA SERVOVÁLVULA

AUTOR: MIGUEL DE GREGORIO SANTAMARINA

DIRECTOR/ES: JAVIER RICO AZAGRA

DEPARTAMENTO: Ingeniería Eléctrica

“DISEÑO, DESARROLLO Y CONSTRUCCIÓN DE SISTEMA DE CONTROL PARA SERVOVÁLVULA”

Peticionario:	Universidad de La Rioja
Autor:	Miguel de Gregorio Santamarina
	Estudiante de Ingeniería Electrónica Industrial y Automática
Director/es:	Javier Rico Azagra
Departamento:	Ingeniería Eléctrica
Lugar:	Logroño
Fecha:	viernes, 6 de julio de 2018

RESUMEN

En este trabajo fin de grado se pretende crear un sistema que sea capaz de controlar la posición de una servoválvula mediante el uso de las placas microcontroladoras *Arduino* y de la herramienta *Matlab & Simulink*. El propósito es observar como con la tecnología actual y con la gran cantidad de plataformas *open source* existentes la programación resulta más sencilla y mucho más barata que años atrás.

La servoválvula a controlar consta de un motor de corriente continua *Crouzet 82861010*, encargado de desplazar la válvula hasta la posición indicada, de un potenciómetro *Bourns* de 10 kΩ multivuelta empleado para medir la posición actual del sistema. La servoválvula, el motor de corriente continua y el potenciómetro tienen relacionados sus comportamientos mediante un conjunto de engranajes que permitirá al sistema llegar al destino deseado.

El control se ha llevado a cabo mediante el empleo de microcontroladores de la familia *Arduino*. Sin embargo, la programación no se realizará directamente desde el *IDE* de *Arduino*, sino que se empleará la herramienta *Matlab & Simulink* para escribir las funciones y comandos necesarios para el proyecto.

La construcción del circuito eléctrico emplea una etapa de potencia capaz de controlar el movimiento del motor. El control de la electrónica de potencia es realizado por el sistema *Arduino,* mediante el empleo de señales *PWM*.

Durante las pruebas de laboratorios se han realizados controles en lazo cerrado que permiten gobernar la velocidad y posición del motor de corriente continua. Por último, se ha realizado el montaje final del sistema con la creación de un circuito integrado (*PCB*), de manera que las tres etapas (potenciómetro, motor, sistema de control) queden conectadas adecuadamente.

Palabras clave:

Motor de corriente continua, servoválvula, Arduino, Driver, microcontrolador, ingeniería de control, lazo cerrado de control de procesos, Matlab & Simulink.

ABSTRACT

The purpose of this project is to create a system with the hability of controlling the behaviour of the servo-valve by using *Arduino* and *Matlab & Simulink*. The main point is to observe how we can program the system easily with the modern technology, one of the best characteristics of the process is that it is cheaper too.

The servo valve system has a DC motor (*Crouzet 82861010*) to be controlled. This engine has the function of moving the valve to the desired point while a potentiometer will indicate us the current position of the system. The servo valve, the DC motor and the potentiometer are joint with a set of gears that will allow the system to reach the objective.

The control will be carried out through *Arduino* targets, but the programming will not be done directly from *IDE Arduino*, *Matlab & Simulink* will be used to write the functions and the commands needed for this project.

The circuit will take a power stage because the DC motor needs an input bigger than 5 volts, it means, the maximum value that *Arduino boards* can work with. So, this will require a *driver* to control the movement of the engine and to make the communication between the board and the motor.

After doing and controlling speed and position, the final assembly of the stage will be carried out and an integrated circuit (*PCB*) will be created to have all components (potentiometer, motor, control system…) properly connected.

Keywords:

Servo valve, DC motor, Arduino boards, Driver, microcontroller, Control Engineering, feedback, closed loop, Matlab & Simulink.

AGRADECIMIENTOS

En este apartado quiero agradecer a todos los que han estado ahí durante este tiempo. En especial a mi padre, mi madre y mi hermana, que me han sufrido los que más y siempre han estado apoyándome.

También quiero agradecer este trabajo a mi compañero de laboratorio David Villota, por esas jornadas intensas que nos pegábamos y por su ayuda, así como a Alejandro Garrido, un amigo que siempre está dispuesto a ayudar y que siempre ha estado disponible cuando le necesitaba. No me puedo olvidar tampoco de Pablo Álvarez, quien en la recta final del proyecto me ha aconsejado y me ha propuesto ideas para la creación de un circuito integrado que cupiese en el poco espacio que se tenía en el sistema.

Gracias a Óscar Ortega Ortiz, quien me ayudó a realizar y construir un circuito integrado desde cero y se preocupó en que todo el procedimiento saliese bien. Gracias también a Sergio Peciña, por contestarme a las dudas que me surgían sobre *Arduino* y por dejarme estar en el *Área UR-Maker* para imprimir las piezas necesarias en 3D.

Gracias a Javier Rico, ya que, sin su ayuda ni su apoyo y, si no me hubiera guiado y aconsejado, no hubiera sido posible llevar a cabo este proyecto de manera satisfactoria.

El ambiente de trabajo estos cuatro meses en la Universidad han sido muy buenos, así como todos los cuatro años que he compartido con unos compañeros de clase que se han convertido en mis amigos. Me llevo amigos y muy buenos recuerdos, no hay nada mejor que eso.

Mención especial tiene ella, mi apoyo, mi pilar, en los momentos de debilidad es cuando más la he buscado y aguantarme no ha sido fácil, pero sin su apoyo esto no tendría sentido.

“Vivir como uno desee: sólo eso merece llamarse éxito.”

"DISEÑO, DESARROLLO Y CONSTRUCCIÓN DE SISTEMA DE CONTROL PARA SERVOVÁLVULA"

DOCUMENTO Nº 1 ÍNDICE GENERAL

Peticionario:	Universidad de La Rioja
Autor:	Miguel de Gregorio Santamarina
	Estudiante de Ingeniería Electrónica Industrial y Automática
Director/es:	Javier Rico Azagra
Departamento:	Ingeniería Eléctrica

ÍNDICE GENERAL

ÍNDICE GENERAL

ÍNDICE GENERAL

ÍNDICE GENERAL

ÍNDICE DE FIGURAS

ÍNDICE GENERAL

ÍNDICE DE TABLAS

"DISEÑO, DESARROLLO Y CONSTRUCCIÓN DE SISTEMA DE CONTROL PARA SERVOVÁLVULA"

DOCUMENTO Nº 2 MEMORIA

Peticionario:	Universidad de La Rioja
Informantes:	Miguel de Gregorio Santamarina
	Estudiante de Ingeniería Electrónica Industrial y Automática
Director/es:	Javier Rico Azagra
Departamento:	Ingeniería Eléctrica

MEMORIA

1. Introducción

1.1. Motivación

En la actualidad las válvulas son una parte fundamental en los sistemas de control de procesos ya que regulan características como el flujo, el volumen, la presión y la dirección de no sólo fluidos, sino también de gases y materiales secos. En el mercado actual es posible encontrar modelos de diversos tamaños y grados de complejidad, también existen muchas razones que motivan a los usuarios a automatizar las válvulas en los procesos industriales.

Algunas de las razones que motivan la constante mejora de los procesas son.

1. La necesidad de centralizar y controlar los procesos industriales.
2. Una forma de prevenir accidentes: los actuadores se usan con frecuencia ante la existencia de un área de una planta peligrosa donde el usuario debe protegerse y proteger también las instalaciones, debido al peligro que encontrará o causará, por ejemplo, al mover las válvulas de manera incorrecta o al crear una chispa en ambientes con presencia de gases explosivos o inflamables.
3. Trabajar fácilmente con una válvula: se motorizan ciertas válvulas porque su tamaño es demasiado grande o porque trabajan a presiones muy altas, por lo que el par es demasiado alto.

Una válvula, sea cual sea su diseño, consiste físicamente en un recipiente que contiene un fluido normalmente presurizado. Un requisito fundamental es que ese fluido no escape al exterior. Por lo que se precisa que el “recipiente” o válvula sea óptimo para contener el fluido. Esto debe ser así tanto como por razón de seguridad y contaminación ambiental como por el coste.

Para automatizar estos procesos, así como todo lo relacionado a procesos industriales que conllevan el uso de maquinaria y, por tanto, riesgo para las personas, se utiliza lo que se conoce como *Ingeniería de Control*, la cual actualmente está siendo desarrollada en gran medida por diversas investigaciones en robótica. Gracias a la *Ingeniería de Control* cosas tan usadas y comunes como el transporte, las comunicaciones y todos los procesos de producción han evolucionado enormemente.

¿Qué es la *Ingeniería de Control*? La *Ingeniería de Control* es la parte de la ingeniería que aplica la teoría de control para diseñar, planificar y desarrollar sistemas con comportamientos deseados. La práctica requiere la utilización de sensores y actuadores de entrada para hacer modificaciones en la respuesta de salida. Se enfoca principalmente en la implementación de sistemas de control a partir de modelamientos matemáticos.

Esta disciplina se ocupó desde sus orígenes de la automatización y del control automático de sistemas complejos, sin intervención humana directa. Campos como el control de procesos, control de sistemas electromecánicos, supervisión y ajuste de controladores y otros donde se aplican teorías y técnicas entre las que se pueden

destacar: Control óptimo, control predictivo, control robusto y control no lineal. Todo ello con trabajos y aplicaciones muy diversas (investigación básica, investigación aplicada, militares, industriales, comerciales, etc.), las cuales han hecho de la *Ingeniería de Control* una materia científica y tecnológica imprescindible hoy en día.

A medida que las plantas modernas van teniendo muchas entradas y salidas se vuelven más y más complejas, haciendo así que la descripción de un sistema de control moderno requiera de una gran cantidad de ecuaciones. Conforme se va aumentando la complejidad, la teoría de control clásica, que trata a los sistemas con una entrada y una salida, pierde su potencial cuando se trabaja con sistemas de entradas y salidas múltiples.

Con la aparición de computadoras digitales en la década de los 60 fue posible el análisis en el dominio del tiempo de sistemas complejos. La teoría de control moderna, basada en el análisis en el dominio del tiempo y en la síntesis a partir de variables de estados, se ha desarrollado para hacer frente a la creciente complejidad de las plantas actuales y los requisitos cada vez más exigentes de precisión, coste y peso en aplicaciones militares, industriales y espaciales.

1.2. Antecedentes

La aparición de los primeros sistemas de control conocidos, ya en la antigüedad, eran mecanismos destinados a controlar caudales para, por ejemplo, regular relojes de agua o controlar el nivel de líquido en una lámpara de aceite o en un recipiente de vino, el cual se mantiene lleno a pesar de los volúmenes que se extraigan de él. De hecho, el control de caudal de fluido se reduce al control del nivel del fluido, ya que una pequeña abertura provocará un caudal constante si la presión es constante.

El mecanismo de control de nivel de líquido inventado en la antigüedad y usado todavía para regular nivel es el de la válvula flotante, que es parecida a la del depósito de agua de un inodoro corriente. El flotador está hecho de manera que cuando el nivel baja, el caudal del depósito aumenta, y cuando el nivel sube, el caudal disminuye y, si es necesario, se corta (*Imagen 1*). En este caso el sensor y el actuador están combinados en el mismo dispositivo, el flotador y la combinación de tubo de alimentación.

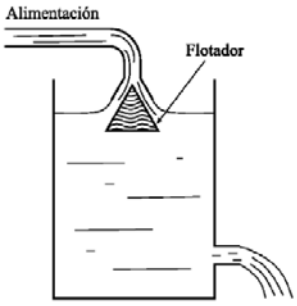


Imagen 1. Combinación flotador con tubo de alimentación

Actualmente se puede llevar a cabo la automatización de los procesos industriales mediante la implementación de instrumentos de medición y control. Estos instrumentos han permitido liberar al operario de su actuación física directa en la planta y, al mismo tiempo, le han permitido la labor única de supervisión y vigilancia del proceso, por lo que ha sido posible fabricar productos complejos con altos estándares de calidad.

La anterior placa implementada en el sistema constaba de una complejidad grande ya que tenía varios microprocesadores que se debían programar mediante interrupciones y funciones típicas para encargarse del control de posición del sistema. Muchos eran los componentes utilizados y el proceso laborioso, pero el sistema no funcionaba cuando se examinó por primera vez.

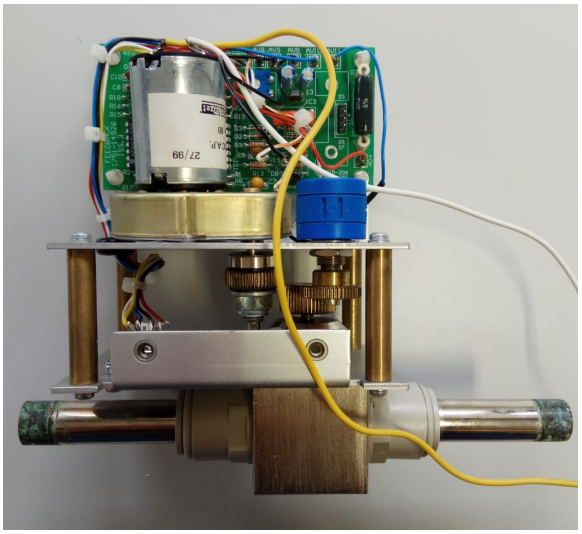


Imagen 2. Imagen de la maqueta anterior

Gracias a la aparición de diversas placas microcontroladoras, como puede ser el caso de las de *Arduino*, se puede programar el sistema fácilmente y desde diferentes plataformas, al tratarse de una herramienta de código libre (*open source)*. Agilizando así el proceso y obteniendo resultados muy buenos. En cuanto al espacio requerido no existe ningún problema ya que los elementos son relativamente pequeños y fácilmente manejables.

Arduino es un microprocesador que nació como un proyecto educativo en el año 2005, por aquel entonces nadie se imaginaba que algunos años más tarde se convertiría en el líder mundial del movimiento *DIY* (*Do It Yourself*).

El nombre del microprocesador procede del bar "*Bar di Re Arduino*" donde Massimo Banzi pasaba algo de tiempo, al mismo tiempo, el nombre del bar viene de un antiguo rey europeo del SXI. El propio Banzi dice que nunca surgió como una idea de negocio, si no como una necesidad de subsistir ante el cierre eminente del Instituto de diseño Interactivo (IVREA) en Italia. Esto significa que al crear un producto "*open hardware*" (de uso público) no podría ser embargado. Hasta la fecha se han vendido más de 300 mil placas en todo el mundo, sin contar las versiones clones compatibles.

Cuando *Arduino* empezó a convertirse en una gran comunidad, su placa comenzó a cambiar para adaptarse a las nuevas necesidades y retos, creando así diferencias en su oferta de placas simples de 8 bits a productos de *IoT*, *weareable*, impresión 3D y entornos embebidos. Todas las placas *Arduino* son de hardware y código abierto, permitiendo así a los usuarios construirlas de manera independiente y adaptarlas a las necesidades particulares.

Gracias a su sencillez y accesibilidad, las placas de *Arduino* han sido usadas en miles de proyectos y aplicaciones. Su software es también sencillo, incluso para

principiantes, y suficientemente flexible todavía para usuarios avanzados. Puede usarse en Mac, Windows y Linux. Profesores y estudiantes lo usan para construir instrumentos científicos baratos, para probar principios físicos y químicos, o para comenzar a iniciarse en el mundo de la programación y la robótica. Diseñadores y arquitectos construyen prototipos interactivos, músicos y artistas lo usan para instalaciones y experimentar con nuevos instrumentos musicales. *Arduino* es la herramienta clave para aprender cosas nuevas. Cualquiera (niños, aficionados, programadores) puede empezar a probar con *Arduino* simplemente siguiendo paso a paso una serie de instrucciones de un kit, o compartir sus ideas en internet con otros miembros de la comunidad *Arduino*.

Existen otros muchos microcontroladores y plataformas disponibles para la computación física. *Parallax Basic Stamp*, *Netmedia's BX-24*, *Phidgets*, *MIT's Handyboard*, y muchos otros ofrecen funcionalidades muy parecidas. Todas estas herramientas toman los detalles problemáticos de la programación del microcontrolador y las agrupan en un paquete fácil de usar.

La tecnología está en constante cambio, simplificando estructuras y mejorando procesos lo máximo posible. Los sistemas de control y adquisición de datos se han convertido en la actualidad en la herramienta más utilizada en las grandes industrias a nivel mundial, transmitiendo información en tiempo real del estado y funcionamiento de equipos y así optimizar las respuestas del sistema.

Además, la tarjeta de adquisición de datos innovada en el proyecto de control de nivel de líquidos conseguirá la comunicación del sistema eléctrico de control y el programa *MATLAB & Simulink*.

1.3. Objeto del trabajo

Diseñar y construir un sistema microcontrolador que reemplace el sistema análogo de control de la servoválvula existente en los módulos de las maquetas *FEEDBACK* de nivel y flujo *38-100*, *38-600*.

1.4. Objetivos

Conforme con el objeto del TFG, existe un objetivo principal que es diseñar un sistema de control para regular el comportamiento de una servoválvula dependiendo de los valores de tensión de entrada que se generen. Para conseguir el objetivo principal del trabajo, se realizará:

- Conocer el funcionamiento de las placas microprocesadoras *Arduino* y su entorno.
- Conocer las posibilidades de manejo del sistema a estudiar mediante la herramienta *Matlab & Simulink*.
- Estudio y obtención del modelo matemático del sistema a través de las leyes físicas y con datos extraídos de las pruebas.
- Diseñar y construir un circuito que se acople a las condiciones que demande el sistema mecánico.
- Elaboración de un sistema en lazo abierto que nos permita darle diferentes entradas al sistema.

- Elaboración de un sistema en lazo cerrado con un controlador de la familia *PID*, que controle de una forma correcta la posición de la servoválvula ante posibles perturbaciones ajenas al sistema.
- Determinar la función de transferencia de la servoválvula en los módulos *FEEDBACK 38-100*, *38-600*.
- Creación de una placa de circuito impreso (*PCB*) para englobar en ella los dispositivos de control y de potencia utilizados.

2. Referencias

2.1. Bibliografía web

- https://es.mathworks.com
- https://tecnoedu.com
- http://www.geocities.ws/calcti/simuv200.html
- http://androminarobot.blogspot.com
- https://es.wikipedia.org
- http://www.fullcustom.es
- https://www.luisllamas.es
- https://www.aprendiendoarduino.com
- https://electronilab.co
- https://www.sparkfun.com

2.2. Libros bibliográficos

- Katsuhiko Ogata. Ingeniería de Control Moderna. Pearson Educación, 2003
- Manual 38-001. Printed in England by FI Ltds. Crowborough

2.3. Artículos bibliográficos

- Juan Diego Munoz Gonzales, Rossy Andrea Lizcano Bohorquez. Diseño y construcción de un circuito electrónico digital y mejoramiento del sistema mecánico de la servoválvula del módulo Feedback de nivel y flujo PROCON 38-001. Trabajo de Grado para optar al título de Ingeniero Electrónico, Universidad Pontificia Bolivariana, Bucaramanga, Colombia, 2009.
- Antonio Pérez Laguarda. Control y supervisión mediante un sistema microcontrolador de los parámetros de calidad de agua de un estanque. Trabajo fin de grado, Universidad de Sevilla, 2007.
- Adrián Antolino Merino. Sistema de adquisición y control de lisímetro de pesada en maceta con Arduino. Trabajo fin de grado, Universidad Politécnica de Cartagena, 2016.
- Ángel Martín Ballesteros, Mario Del Río Carbajo. Control de posición de un balancín con Arduino. Trabajo fin de grado, Universidad de Valladolid, 2013.

3. Definiciones y abreviaturas

- Buffer: en informática, un buffer de datos es un espacio de la memoria en un disco o en un instrumento digital reservado para el almacenamiento temporal de información digital, mientras que está esperando ser procesado. En electrónica, un amplificador buffer es un dispositivo que acopla impedancias en un circuito.

- Matlab: Herramienta de software matemático que ofrece un entorno de desarrollo integrado (IDE) con un lenguaje de programación propio (lenguaje M).

- Simulink: Entorno de programación visual, que funciona sobre el entorno de programación Matlab.

- RPM: revoluciones por minuto

- Open source: plataformas de código abierto gratis y accesibles para cualquier usuario que se quiera introducir y profundizar en el entorno de programación ofrecido. También poseen la opción de, al ser código abierto, ser mejoradas y desarrolladas por cualquier persona.

- PWM: Viene del inglés *Pulse Width Modulation*, cuyo significado es modulación por ancho de pulsos en español. Está referida a la acción de control que se le envía al motor.

- Ciclo de trabajo: en inglés "*Duty cycle*", es la relación existente entre el tiempo en el que la señal se encuentra en estado activo, a nivel alto, y el período de la misma.

- Toolbox: entorno gráfico que permite introducir la programación de ordenadores y dispositivos en ámbitos sin competencias informáticas. Parte de la premisa de que la persona, al resolver un problema, realiza cálculos en una secuencia determinada (es decir, procede algorítmicamente) y puede expresarlos en un lenguaje informático sencillo, del mismo modo que lo hace en estilo libre sobre un cuaderno o pizarra.

- PCB: siglas en inglés de placa de circuito impreso (*Printed Circuit Board).*

- TFG: Siglas de Trabajo Fin de Grado.

4. Desarrollo hardware

4.1. Elección del microcontrolador

En la actualidad existen inmensidad de microcontroladores en el mercado capaces de soportar el proyecto que se va a realizar. El objetivo que se busca es que la plataforma sea de código libre, para facilitar el desarrollo del trabajo. Entre estas plataformas se encuentran, entre otras, *Raspberry Pi*, *BeagleBone, Sharks Cove*, *Waspmote* o *Arduino*.

En este proyecto se ha elegido la plataforma *Arduino* debido a que, además de simplificar todo el proceso de trabajo con microcontroladores, ofrece una serie de ventajas que le diferencia respecto a otros sistemas.

- Precio: las placas *Arduino* son relativamente baratas en comparación con otras plataformas microcontroladores.
- Multiplataforma: El software *Arduino* (IDE) puede ejecutarse en sistemas operativos de Windows, Mac y Linux. Mientras que la gran mayoría de los sistemas están limitados a Windows.
- Entorno de programación simple y claro: El software *Arduino* (IDE) es sencillo de usar para gente sin experiencia previa, y aún lo suficientemente flexible para que los usuarios más avanzados puedan sacarle utilidad.
- Código libre y software extensible: el software de *Arduino* está publicado como herramienta de código abierto, disponibles para la extensión por programadores experimentados. El lenguaje puede ser expandido a través de bibliotecas C++. Para las personas que quieran entender los detalles técnicos pueden realizar el salto de *Arduino* al lenguaje de programación AVR-C en el cual está basado. También existe la posibilidad de agregar código AVR-C directamente en los programas *Arduino*.
- Código libre y hardware extensible. Los diseños de las placas *Arduino* están publicados bajo una licencia de Creaciones Comunes, por tanto, experimentados diseñadores de circuitos pueden realizar su propia versión de la placa, ampliando y mejorando las características dependiendo de su posterior uso.

4.2. Arduino

Arduino es una plataforma electrónica de código abierto (*open source*) basada en un manejo sencillo tanto de software como hardware. Las placas *Arduino* son capaces de leer entradas, como puede ser la temperatura mediante un sensor, y convertirlas a salidas que permitan activar un ventilador o encender un LED. Otros ejemplos de aplicaciones son el control de posición y velocidad de motores mediante otro tipo de sensores. Se le puede decir a la placa lo que se desea que haga mediante el envío de instrucciones al microcontrolador de la placa. Para ello se utiliza el lenguaje de programación *Arduino*, basado en *Wiring* y el software *Arduino* (IDE) basado en *Processing*.

Desde su creación, en 2005, *Arduino* ha sido utilizado en miles de proyectos, desde objetos cotidianos a complejos instrumentos científicos. Diseñadores alrededor del mundo (estudiantes, aficionados, programadores, profesionales) se han unido a esta

plataforma de código abierto, sus contribuciones han proporcionado una gran cantidad de conocimiento que puede ser tanto para novatos como para expertos.

4.2.1. Placas de Arduino

La variedad de productos ofrecidos por *Arduino* es enorme, ofrece placas, módulos, accesorios y kits enfocados a una aplicación concreta. Existen placas pensadas para personas poco o nada experimentadas en el mundo de *Arduino* y la programación de microcontroladores, placas con características un poco más avanzadas. *Arduino* orientado a *IoT*, pensados para llevar encima (*weareable*), o incluso otros pensados exclusivamente para el manejo de impresoras 3D.

A continuación, se va a comentar brevemente las características más importantes de algunas de las placas *Arduino*.

- *Arduino UNO*: esta es la placa más conocida de *Arduino* y la más recomendada para comenzar. Ésta fue la primera en salir al mercado, el resto de las placas posteriores están apoyadas en ella para su diseño. El microcontrolador que tiene es el ATmega320 de 8 bits a 16 Mhz con una alimentación de 5V. Uno de sus defectos es la memoria, la cual es algo limitada, pero no impide que no sea compatible con multitud de proyectos. La placa consta de 14 pines digitales, de los cuales 6 de ellos pueden usarse como *PWM* (los que tienen el símbolo de la *virgulilla "~"),* y 6 pines analógicos. Los pines pueden trabajar con corrientes de hasta 40 mA.

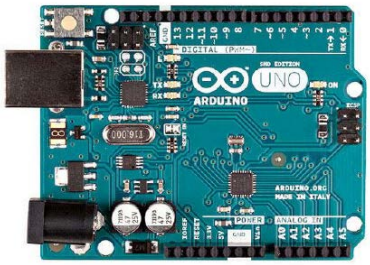


Imagen 3. Placa Arduino UNO

- *Arduino Leonardo*: Es muy similar a *Arduino UNO*. Sus diferencias son su tamaño más reducido (usa sólo conexión USB), el número de pines (12 pines analógicos y 20 digitales), y que los pines son sólo perforaciones en la placa, no cuentan con las tiras de pines para la conexión. También se diferencian en el microcontrolador, el cual es diferente, ATmega32u4.

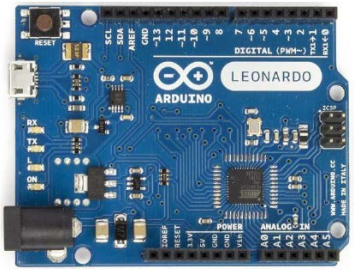


Imagen 4. Placa Arduino Leonardo

- *Arduino Mega*: El microcontrolador que usa es el ATmega2560, con capacidades superiores al *Arduino UNO*. Una de las principales características de *Arduino Mega* es su gran cantidad de pines. Posee 54 pines digitales, de los cuales 15 de ellos son *PWM*, y 16 pines analógicos.

Imagen 5. Placa Arduino Mega

- *Arduino Due*: Es la primera placa basada en un microcontrolador ARM de 32 bits, exactamente el Atmel SAM3X8E ARM Cortex-M3. A diferencia de la mayoría de las placas *Arduino*, maneja un voltaje de 3,3 voltios. Está orientada a proyectos con alta capacidad de procesamiento. Posee 54 pines digitales, 12 de ellos *PWM* y 12 analógicos.

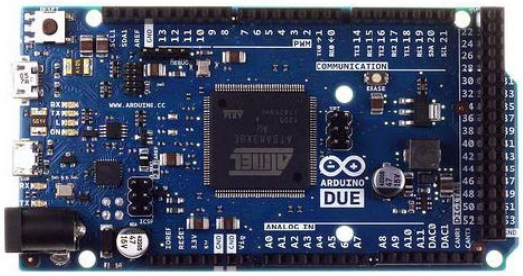


Imagen 6. Placa Arduino Due

- *Arduino Nano:* placa mucho más pequeña que las explicadas anteriormente. Basada en el microcontrolador ATmega328. Tiene una entrada mini-usb por la que se puede subir el código fuente para la ejecución de los comandos. Posee 14 pines digitales, 8 pines analógicos, una memoria de 16 KB, 1 KB de SRAM y 512 bytes de EPROM. Su ClockSpeed es de 16 MHz, funciona con un voltaje que puede estar en el rango de 7 a 12 voltios. La corriente que entrega es de 40 mA. Salvo algunas desventajas como el menor número de puertos, es muy parecido al *Arduino UNO*.

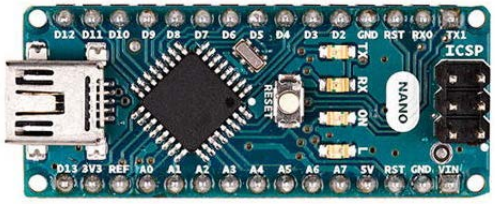


Imagen 7. Arduino Nano

4.3. Salida PWM

Lo más habitual es usar entradas digitales y analógicas para recibir señales del mundo. Así como también se puede interactuar con el entorno a través de salidas digitales.

Pero en ocasiones no es suficiente con una señal digital (*ON/OFF*), si no que se necesita proporcionar un valor analógico de tensión para realizar diversas tareas como regular la intensidad de iluminación de un LED, o, como en este caso, variar la velocidad del motor de corriente continua. La salida *PWM* sirve para emular una señal analógica de tensión desde *Arduino*.

Las salidas analógicas, como pasa con las entradas del mismo tipo, son más complicadas que las digitales. La mayoría de los automatismos (en este caso, *Arduino*) no son capaces de proporcionar una auténtica salida analógica, ni siquiera son capaces de proporcionar una salida analógica discretizada (es decir, a saltos) de tensión. Lo único que pueden suministrar es una salida digital de -Vcc a Vcc (por ejemplo, 0V y 5V).

Una solución para salvar esta limitación y simular una salida analógica consiste en activar una salida digital durante un tiempo y mantenerla apagada durante el resto. El valor medio de la tensión de salida, a lo largo del tiempo, será igual al valor analógico deseado.

Existen diferentes formas de realizar esta aproximación, una de las más sencillas y utilizadas es la que se llama *modulación por ancho de pulso (PWM).* Durante esta acción se mantiene constante la frecuencia (es decir, el tiempo entre el disparo de los pulsos), mientras que se hace variar la anchura del pulso.

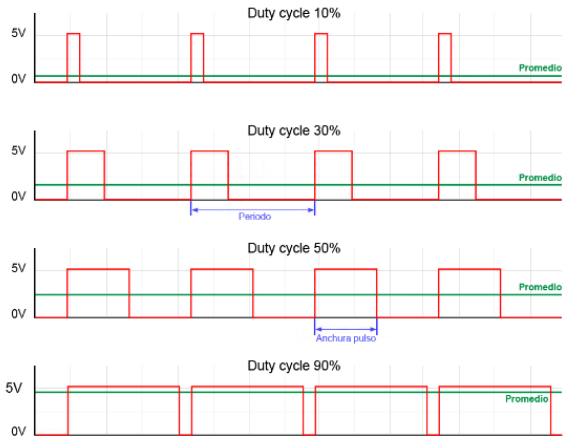


Imagen 8. Señal PWM

La cantidad de tiempo en la que la señal está en valor alto (encendida), respecto al total del ciclo, se denomina *"Duty cycle",* y se suele expresar en tanto por ciento.

Calcular la señal promedio es algo inmediato, consiste en el producto de la tensión máxima y el *Duty Cycle*, tal y como se muestra en la siguiente expresión.

$$V_{medio} = (V_{CC+} - V_{CC-}) \cdot \frac{Duty\ Cycle}{100} \quad (4.1)$$

Pero en una salida *PWM* el valor de tensión es realmente Vcc. Por ejemplo, si se alimenta un dispositivo que requiere 3V mediante este tipo de señal, se le está suministrando en realidad 5V, pero durante el 60% del tiempo y 0V durante el 40%. Un aspecto muy a tener en cuenta es que, si este dispositivo soporta como máximo 3V, se puede dañar si se alimenta con señales *PWM* de 5V.

La señal *PWM* es suficiente para emular una señal analógica en muchas aplicaciones. Por ejemplo, se puede variar la velocidad de un motor de corriente continua con la señal pulsada, en este caso la inercia del motor se encargará de que el efecto que tiene la *PWM* sea despreciable. Pero en función de la frecuencia se pueden apreciar vibraciones o ruidos, en cuyo caso se debería variar la frecuencia de la señal.

Por otro lado, se deben tener en cuenta las repercusiones que suponen la rápida conexión y desconexión de la señal pulsada en el dispositivo alimentado.

En las placas *Arduino* las salidas *PWM* aparecen identificadas con el símbolo "~" junto al número del pin. En *Arduino UNO* se disponen de 6 salidas *PWM* de 8 bits en los pines digitales 3, 5, 6, 9, 10 y 11.

5. Circuito eléctrico para el control de velocidad

En el laboratorio se procedió a realizar un montaje con el que se pudiese realizar el control de velocidad del motor de corriente continua que hace la función de actuador en el sistema de servoválvula.

5.1. Componentes

5.1.1. Motor Crouzet 82861010

Este motor empleado en la servoválvula recoge las siguientes características generales.

CARACTERÍSTICAS GENERALES	
Tipo	828610
Voltaje	12 V
Velocidad estándar (rpm)	4300
Velocidad de salida (rpm)	54
Ratio de la reductora (i)	80
Motor	828600
Reductora	810210
Máximo par permitido por la reductora bajo unas condiciones continuas para 1 millón de vueltas	0.5
Carga estática axial (daN)	1
Carga estática radial (daN)	8
Máxima potencia de salida (W)	3.9
Potencia nominal de salida (W)	3
Aumento de temperatura de la reductora(ºC)	50
Peso (g)	160

Tabla 1. Características del motor 82861010

Imagen 9. Motor Crouzet 82861010

5.1.2. Driver L298N controlador para motores DC y paso a paso con Arduino

El dispositivo está basado en el chip *L298N*, permite controlar dos motores de corriente continua o también un motor paso a paso bipolar hasta 2 amperios.

El módulo cuenta con todos los componentes necesarios para funcionar sin necesidad de elementos adicionales. Está compuesto por unos diodos de protección y un regulador *LM7805* que suministra 5V a la parte lógica del integrado *L298N*. Cuenta también con jumpers de selección para habilitar cada una de las salidas del módulo (A y B). La salida A la forman *OUT1* y *OUT2* y la salida B por *OUT3* y *OUT4*. Los pines de habilitación son *ENA* y *ENB* respectivamente.

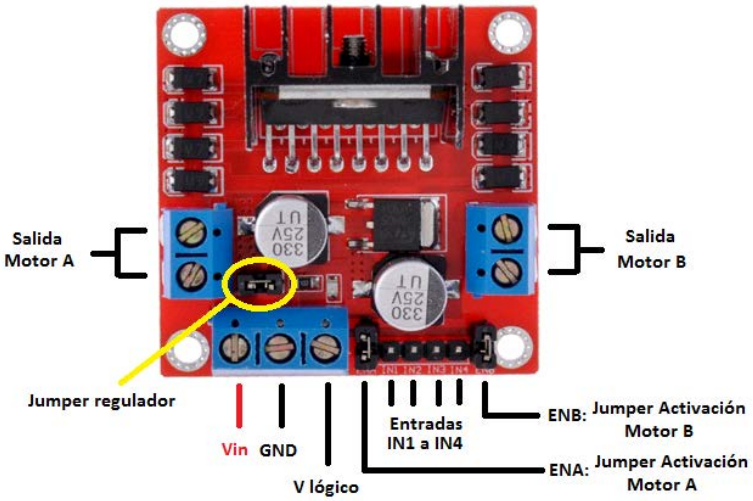


Imagen 10. Driver L298N

Cuando se activa el jumper regulador de 5 voltios, es decir, cuando tiene puesto la muesca protectora, el módulo permite una alimentación de entre 6 y 12 voltios en corriente continua. Como está activo, el pin marcado como *V lógico* tendrá un voltaje de 5 voltios DC. Dicho voltaje se puede utilizar para alimentar la parte de control del módulo, ya sea un microcontrolador o un *Arduino*, es recomendable que el consumo no sea mayor a 500 mA.

Si se desactiva dicho jumper (extrayendo de sus patillas la muesca protectora), el módulo permite una alimentación de 12 voltios a 35 voltios DC. Como el regulador en este caso no está funcionando, se debe conectar el pin de +5 voltios a una tensión de 5 voltios para alimentar la parte lógica del *L298N*. Generalmente la tensión es la misma que la de la parte de control, ya sea un microcontrolador o *Arduino*.

5.1.3. Encoder FC-03

El encoder *FC-03* o encoder *FZ0888* es un módulo sensor de velocidad de infrarrojos con el comparador *LM393*) u optointerruptor. Gracias a él se puede calcular la velocidad de rotación, pudiéndose usar como interruptor óptico si se coloca una corona dentada que gire unida al eje del motor.

El sensor funciona de la siguiente manera. Si se hace pasar cualquier cosa entre la ranura del sensor, éste genera un pulso digital en el pin D0. Dicho pulso va de 0V a 5V y es una señal digital *TTL*. Luego con *Arduino* es posible leer este pulso y transformarlo a una señal analógica con las salidas *PWM* que tiene incorporadas la placa.

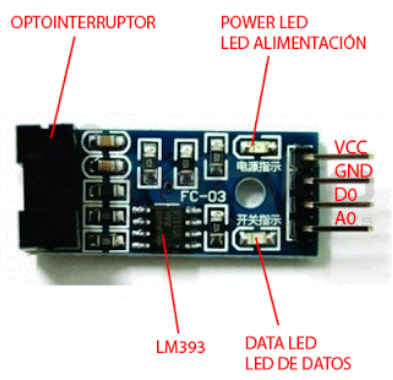


Imagen 11. Encoder FC-03

Los pines de conexión del encoder son:

- VCC: alimentación del módulo de 3.3V a 12V.
- GND: tierra.
- D0: señal digital de los pulsos de salida.
- A0: señal analógica de los pulsos de salida. Señal de salida en tiempo real (normalmente no se usa).

Principales características técnicas:

- Dimensiones: 32 x 14 x 7 mm.
- La ranura de lectura del sensor tiene un ancho de 5 mm.
- Dos salidas, una digital y otra analógica.
- LED indicador de alimentación.
- LED indicador de los pulsos de salida del pin D0.

5.1.4. Corona circular para contar pulsos del encoder

Para poder contar los pulsos del encoder óptico fue necesario crear una corona circular capaz de cortar el haz de luz. Para ello se realizaron las medidas pertinentes tanto del eje del motor, para acoplarle la corona, como de la distancia que había hasta la posición del encoder que se tenía. Una vez realizadas las medidas, se pasó a crear la corona. *ThingIverse* es una web donde se recoge una gran diversidad de dispositivos que se pueden imprimir en una impresora 3D. En este caso se buscó una rueda de encoder parametrizable, de tal modo que se pudiese diseñar y configurar los parámetros necesarios.

Configuración corona circular	
Diámetro exterior (mm)	52
Altura de la rueda (mm)	2.5
Longitud de las ranuras (mm)	12
Número de ranuras	36
Relación entre ranuras y sección sólida entra ellas (0.5 es igual)	0.5
Diferencia diámetro exterior a aberturas (mm)	1.75
Diámetro espacio eje del motor (mm)	5
Hueco atraviesa la corona completamente	Sí

Tabla 2. Características rueda dentada

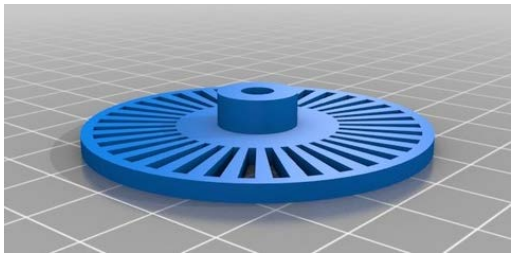


Imagen 12. Archivo .stl de la corona circular

Una vez creado el archivo *".stl"* que permite la creación de la pieza en la impresora se procede a su realización en el *"Área UR-Maker"*.

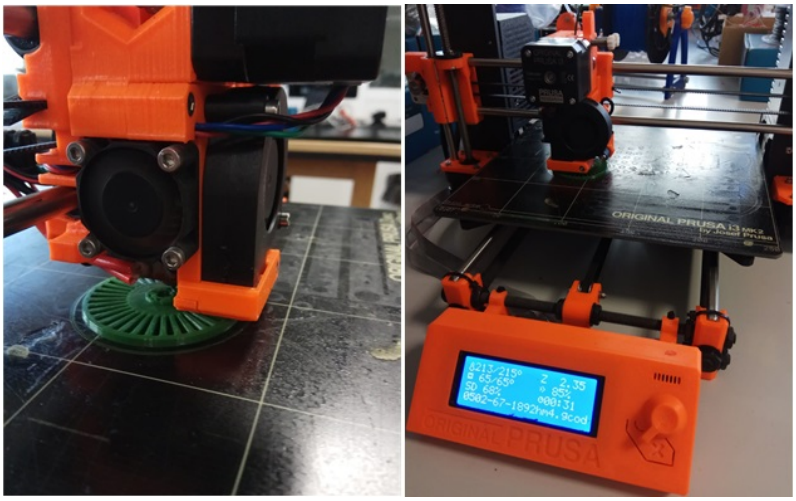


Imagen 13. Impresión de la corona circular

5.1.1. Soporte del encoder

Una vez colocada la corona circular en el eje de salida del motor tras la reductora se observó que había que modificar la posición inicial pensada para que los pulsos fuesen fiables y el encoder *FC-03* estuviese en un puesto fijo y a una altura superior a la inicial. Por lo que se creó un soporte con la herramienta *AutoCAD*, cuyo fichero se guardó en la versión *".stl"* para poder ser imprimido con una de las impresoras 3D que se encuentran en el *Área UR-Maker*.

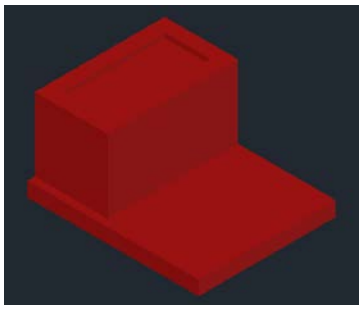


Imagen 14. Diseño en AutoCAD del soporte del Encoder

5.2. Montaje del circuito

Conocido el funcionamiento de cada uno de los componentes que forman parte del circuito para controlar la velocidad, se procede a realizar su montaje. Mediante el programa realizado en *Simulink* para *Arduino* se controla todo el funcionamiento. En este caso también se ha utilizado una *protoboard* para que el montaje fuese más claro y seguro. Como puede observarse se ha utilizado una fuente de corriente continua para generar esos 12 voltios que necesita el motor para moverse.

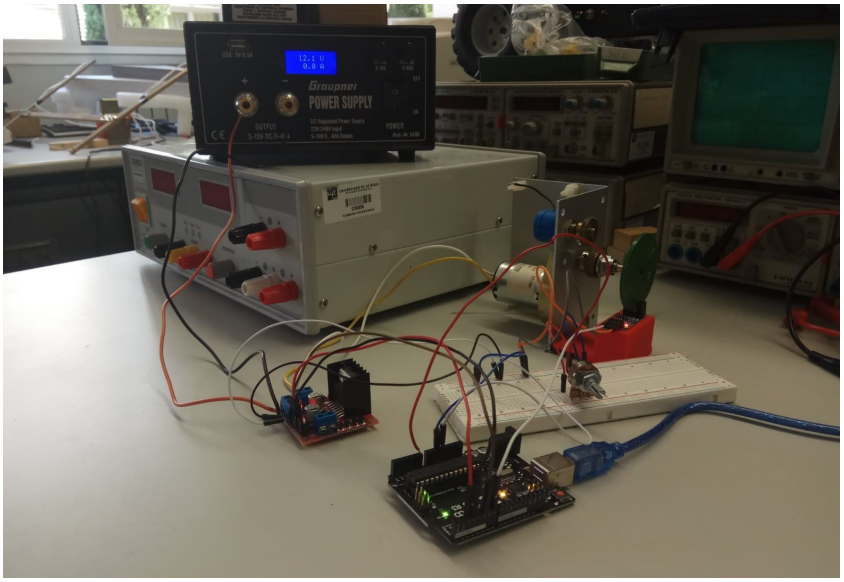


Imagen 15. Montaje circuito eléctrico para el control de velocidad

6. Circuito eléctrico para el control de posición

Para el control de posición de la servoválvula no es necesario usar el encoder óptico ya que no hace falta medir la velocidad del motor en su eje, tampoco se utilizan la corona circular ni el soporte del encoder que ayudaban a tomar los pulsos.

Lo que sí que se introduce es el potenciómetro que está en la maqueta de la válvula y que es solidario al movimiento del motor, servirá para indicar la posición del sistema en cada momento.

6.1. Componentes

Como ya se ha comentado, se utilizan los elementos empleados para el control de velocidad menos aquellos que se usaban para medir dicha característica. Es decir, se utiliza en motor *Crouzet 82861010* y el driver *L298N,* así como también el *Arduino*. A este modelo se añadirá el potenciómetro citado.

6.1.1. Potenciómetro multivuelta Bourns 10kΩ ± 5%

En la maqueta de la servoválvula se usa un potenciómetro como elemento de referencia para realizar el control de posición. Se trata de un potenciómetro de la marca *Bourns* multivuelta, de 10kΩ. Este potenciómetro alcanza el valor final una vez giradas 10 vueltas en un sentido y el valor inicial si se le vuelve a girar esas 10 vueltas en sentido contrario.

Imagen 16. Potenciómetro multivuelta

Este elemento sirve para poder tener una referencia del movimiento del motor en cada momento y conforme a ello el control de posición realizará lo necesario para colocar el sistema en las nuevas posiciones que se marquen a la entrada.

6.2. Montaje del circuito

Para probar el control de posición se usa un potenciómetro como entrada al sistema para observar si el sistema responde bien a los cambios introducidos. La apertura de la válvula se coloca en su punto medio, así como la posición del potenciómetro, es decir, si es de 10 vueltas se implementa en el sistema con 5 vueltas dadas.

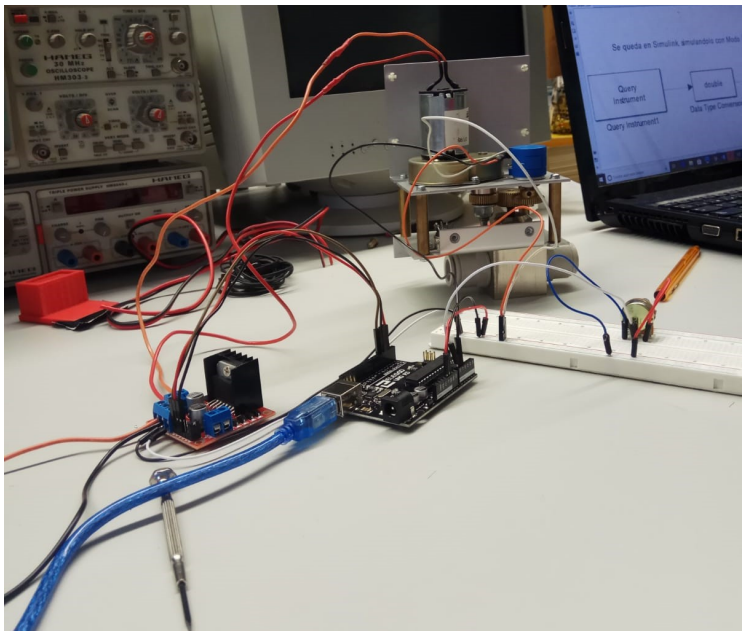


Imagen 17. Montaje circuito eléctrico para el control de posición

7. Programación para control de la servoválvula

El primer requisito para diseñar e implementar sistemas de control es conocer las características de la planta que se desea controlar, esto implica conocer el modelo matemático que la represente. En el control clásico, la función de transferencia es la representación matemática que describe el modelo del sistema y que permite establecer su valor de salida ante una determinada entrada o perturbación. Gracias a la función de transferencia es posible determinar la estabilidad, los errores existentes ante perturbaciones externas y los parámetros que se desean corregir. Pero no siempre es posible establecer la función de transferencia a menos que se conozcan de antemano los parámetros físicos de la planta que se pretende controlar.

En cada caso la obtención de modelos es específica, por ejemplo, en este trabajo se discuten los resultados al modelar un motor de corriente continua con un encoder óptico.

El software utilizado para analizar e implementar las diferentes leyes de control es *Matlab & Simulink.* Gracias a la *toolbox "Simulink Support Package for Arduino Hardware"* se permite programar la placa *Arduino* sin necesidad de usar el entorno *IDE de Arduino.* Para el control de velocidad del motor y de posición de la servoválvula se van a realizar circuitos de control de lazo cerrado, los cuales permiten comparar la entrada con la salida y corregir la diferencia existente entre ellos.

El motor *Crouzet 82861010,* como se aprecia en las características mostradas en el apartado de componentes, tiene una velocidad máxima de 4300 revoluciones por minuto y posee una reductora acoplada al eje de salida con una relación de 1/80.

Para estudiar el motor, la servoválvula y sus respuestas, los programas que se realizan son diversos. En primer lugar, se procede a averiguar la estática del motor, que

sirve para determinar la ganancia del actuador para utilizarla en cálculos posteriores, también se va a poder conocer su dinámica.

En cada apartado se crean dos archivos diferentes pero complementarios. El primero tiene todo el código necesario a transferir a *Arduino*, dicho código se transmite a la placa desde el botón "*Deploy to hardware*", una vez subido no dejará hacer ninguna modificación. Es decir, si se desea cambiar o añadir algo más, es necesario volver a transferir el código a la placa una vez hecha la modificación. El código subido a la placa tampoco permite visualizar los valores obtenidos tras la ejecución. Para permitir el intercambio de datos entre PC y *Arduino* existen dos bloques que realizan la transmisión y recepción de datos por el puerto serie.

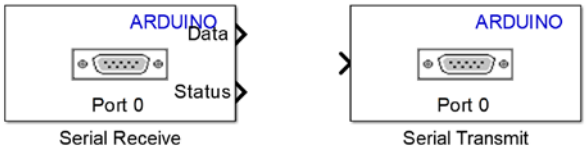


Imagen 18. Bloques comunicación puerto serie PC-Arduino

Estos bloques permiten tanto la recepción como la transmisión de datos desde *Arduino*, es decir, la placa procesa los valores que recibe del PC o de otro instrumento y los transmite de la misma manera, pero todo en la placa, ambas estructuras trabajan con valores en formato *uint8*, es decir, se reciben y se transmiten números enteros de 8 bits sin signo.

Para realizar lo mismo desde el PC se crea otro programa que se ejecuta de manera Normal, sin transferirlo a la placa en este caso, sirve para que se pueda modificar la entrada como se desee y así observar el comportamiento del sistema. También permite recibir y analizar la información procedente de *Arduino*.

En este caso se tienen otros dos bloques, con la función comentada de recibir y recopilar datos de *Arduino*.

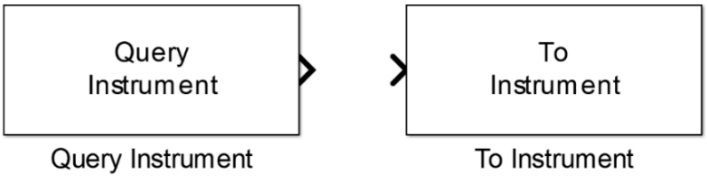


Imagen 19. Bloques comunicación puerto serie Arduino-PC

Estos bloques permiten una mayor configuración que los anteriores, recibiendo y transmitiendo datos en diferentes formatos (ASCII, binario), además de poder modificar parámetros como las muestras leídas cada período o la velocidad de transmisión y recepción. Como los bloques que trabajan desde *Arduino* lo hacen con datos *uint8*, es aconsejable que, en este caso, se haga lo mismo también.

Para terminar este apartado, se debe comentar que todo el procedimiento trabaja con un tiempo de muestreo de 0.01 segundos, valor que permite obtener unos resultados precisos pero que presenta cierto ruido. Con este tiempo de muestreo el experimento corre en tiempo real, es decir, un segundo en simulación corresponde con un segundo en la realidad.

7.1. Estática

La estática del motor de corriente continua es empleada para averiguar la ganancia que tiene el sistema. Además, es de gran utilidad para comprobar que, efectivamente, el motor corre a la velocidad determinada por la hoja de características (54 rpm).

Para su estudio se emplean diferentes *scripts* o archivos en *Simulink*. *Simulink* permite, a partir de bloques y diagramas de flujo, configurar el código necesario que introducir a la placa, convirtiendo a código *C++* los bloques y las diferentes estructuras creadas.

Como se ha explicado, un archivo se genera y se transfiere a *Arduino* mientras que el otro sirve para modificar la entrada y obtener los valores de salida. Se ha optado por esta opción ya que en la comunicación del ordenador con *Arduino* siempre se pierde tiempo y precisión, por lo que cuantos menos bloques y valores se transmitan y reciban en la comunicación menos retraso y mayor precisión existe durante el proceso. De esta manera el programa principal corre desde *Arduino*, introduciéndole el código al microcontrolador de la placa, estando sólo en el *script* que se ejecuta desde el ordenador los valores de entrada y la recepción de la respuesta.

7.1.1. Estática Arduino

Se llama *Estática Arduino* al programa que se ejecuta en la placa. Tiene todo lo necesario para tratar la señal de entrada y procesar la respuesta de salida del motor.

Dicho archivo va a recibir los valores de entrada del *script* generado que corre desde *Simulink* para determinar la velocidad del motor.

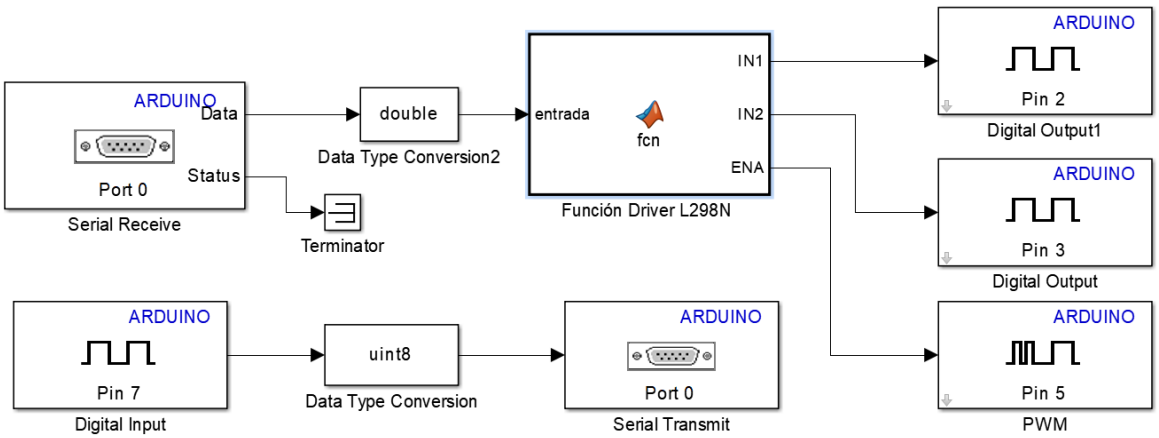


Imagen 20. Programa Estática Arduino

En este caso se crea una función de *Matlab* en el propio entorno de *Simulink*, dicha función es nombrada como "*Función Driver L298N*" y sirve para indicar el sentido de giro del motor y la velocidad que debe de llevar dependiendo de la entrada. Como la comunicación tanto de PC con *Arduino* como de *Arduino* con PC trabaja con valores enteros sin signo de 8 bits, no se va a poder trabajar con valores negativos en un principio, por lo que se convierten en positivos todos los datos de entrada antes de ser transferidos. Es decir, como en el caso de la estática los datos van a ir de -100 a 100 representando el porcentaje de potencia en valores *PWM*, basta con sumarles 100 para que todos sean

positivos. De esta manera, los valores negativos van de 0 a 100 y los positivos de 100 a 200.

Para velocidad hablamos de datos *PWM* (modulación por ancho de pulso), dichos datos dan una salida u otra dependiendo del ciclo de trabajo en cada caso. En la placa *Arduino* se pueden tratar señales *PWM* desde los pines digitales que tengan una virgulilla "~". De esta manera se consigue que el motor corra en todas las velocidades posibles, asociando un valor 0 de señal cuando la entrada es 0, y unos valores máximos a cada sentido cuando la señal de entrada es -100 y 100 respectivamente.

La función creada es la siguiente.

```
function [IN1, IN2, ENA] = fcn(entrada)
%#codegen

if entrada > 100;                   % entrada mayor que 100(valores +)
    IN1 = 0;                        % parámetros dirección del motor
    IN2 = 1;                        % sentido horario: IN1=0, IN2=1
    ENA = (entrada-100)*2.55;       % se introduce el valor PWM,

else if entrada < 100;              % entrada menor que 100 (valores -)
    IN1 = 1;                        % parámetros dirección del motor
    IN2 = 0;                        % sentido anti-horario
    ENA = abs(entrada-100)*2.55;    % se introduce el valor PWM,
                                      muestra la velocidad del motor
else
    IN1 = 0;                        % si entrada es cero, el motor permanecerá parado
    IN2 = 0;                        % no habrá señal en las entradas
    ENA = 0;
    end
end
end
```

La entrada digital procedente del pin 7 de la placa de *Arduino* son los pulsos que se cuentan del encoder. Es decir, de cuando la corona circular corta el haz de luz del encoder óptico. Como nuestra corona va a tener 36 agujeros, va a cortar el haz de luz 36 veces por vuelta. Pudiendo sacar así el valor de velocidad del motor, el cual se calcula en el *script* hecho para trabajar en el PC. En dicho archivo, como se puede observar, se transmiten los pulsos del encoder, convirtiéndolos antes en formato *uint8.*

7.1.2. Estática Simulink

Para calcular los valores de velocidad se genera el siguiente código en *Simulink*. Tomando como valores de entrada los de una escalera cuyo código se crea en un *script* de *Matlab.* Los valores de entrada pueden ser los que se quieran entre -100 y 100. Como se puede observar y se ha comentado, antes de transmitirlos al instrumento, en este caso *Arduino*, se les suma un 100 para que todos ellos sean positivos y se transfieran tal y como se quiere, de esta forma se aprecia con claridad el comportamiento del motor en los diferentes sentidos de giro.

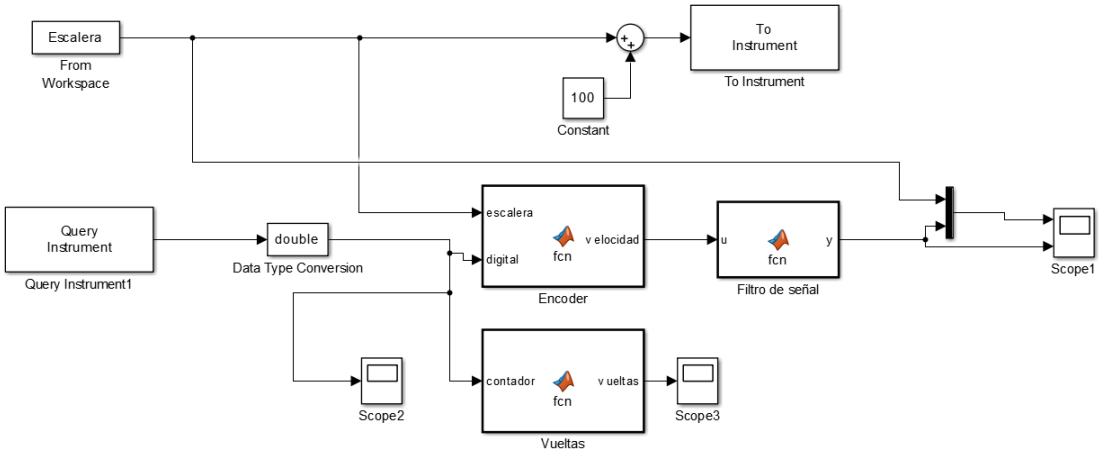


Imagen 21. Programa Estática Simulink

En este programa se ven otra vez diferentes bloques de funciones de *Matlab*, la que tiene por nombre *Encoder* sirve para transformar los pulsos obtenidos del encoder óptico en valores de velocidad. Como se ha explicado, la corona circular irá interrumpiendo el haz de luz, dando pulsos cada vez y, conocidos el número de pulsos obtenidos en un determinado tiempo, es posible calcular la velocidad.

La variación de ángulo entre pulso y pulso se calcula teniendo en cuenta que la corona circular creada tiene 36 huecos o aberturas:

$$\Delta\theta = \frac{°/vuelta}{aberturas/vuelta} = \frac{360°/vuelta}{36\ aberturas/vuelta} = 10° \qquad (7.1)$$

El encoder va a ir detectando interrupciones, cuando la carona circular corte el haz de luz, es decir, cuando la superficie sin hueco lo cruce, va a detectar un 0, mientras que cuando una abertura pase por el haz de luz sin cortarla se asigna un valor de 1, nivel alto.

```
function velocidad  = fcn(escalera, digital)

Atheta=10; % variación del ángulo entre paso y paso
Ts=0.01;   % tiempo de muestreo

persistent t_anterior u_anterior tiempo V
if isempty(tiempo)
    u_anterior=0; % señal anterior
    t_anterior=0; % tiempo anterior antes de que dé el próximo flanco
    tiempo=0;     % tiempo actual
    V=0;          % velocidad
end

if (u_anterior==0 && digital==1 && escalera > 0)
    V=Atheta/(tiempo-t_anterior);
    t_anterior=tiempo;
else if (u_anterior==0 && digital==1 && escalera < 0)
    V=-Atheta/(tiempo-t_anterior);
    t_anterior=tiempo;
    end
end
if (escalera == 0)
    V=0;
end
u_anterior = digital;
tiempo = tiempo+Ts;
velocidad = V*60/360;
```

Como hay 36 huecos, el encoder va a detectar un nuevo pulso 10º después de haberse producido el anterior. Por lo que hay pulsos cada 10º de giro de la rueda.

En esta función, si la variable tiempo está a cero, es decir, si todavía no se ha ejecutado el programa, las demás variables serán cero también. En cuanto se inicie el proceso y el encoder detecte un flanco ascendente, tras un flanco descendente, divide el ángulo girado entre paso y paso por el tiempo transcurrido y saca la velocidad en grados por segundo.

Si el valor de la entrada al sistema, en este caso el valor de la *escalera*, es nulo, la velocidad es cero también porque el motor está parado. Una vez realizado este proceso se asignan a la salida el valor de velocidad convertido a revoluciones por minuto ya que es una opción más visible y clara del comportamiento del motor. Luego se asignan a las otras variables los valores siguientes para volver a entrar al bucle.

La función que está a continuación de la del *Encoder* es un filtro que depura la señal y la limpia. El filtro consiste en realizar una media entre los valores obtenidos para así conseguir eliminar parte del ruido que aparece en la señal.

```
function y  = fcn(u)

persistent u_an;
if isempty(u_an)
    u_an=[0 0 0 0];
end

u_an(2:end)=u_an(1:end-1);
u_an(1)=u;
y=sum(u_an)/length(u_an);
```

La última función que aparece en el archivo (*Vueltas*) sirve para comprobar que los pulsos que se leen del encoder óptico son buenos y que realmente el motor da las vueltas que debe dar dependiendo de los valores de entrada. Por ejemplo, se comprobó que el motor daba 54 revoluciones a máxima velocidad en un minuto.

```
function vueltas  = fcn(contador)

persistent pulsos u_anterior
if isempty(pulsos)
    u_anterior=0;
    pulsos=0;           % pulsos
    vueltas=0;          % vueltas
end

if (u_anterior==0 && contador==1)
    pulsos=pulsos+1;
end

u_anterior = contador;
vueltas= pulsos/36;
```

Con todo este procedimiento se pueden obtener unos valores de la estática del motor óptimo para operar con ellos en sucesivos procesos. Los valores están mostrados en el apartado donde se analizan todas las respuestas recogidas.

7.2. Control de velocidad

Una vez conocida la estática del motor y su función de transferencia es posible realizar su control de velocidad. Dicho control se va a llevar a cabo con funciones iguales o muy parecidas a la de la estática. La diferencia está en los lazos de control cerrados que se implementan ahora, para la estática no hacía falta cerrar el lazo porque lo que interesaba era ver el comportamiento del motor en lazo abierto ante una entrada cualquiera.

En el control de velocidad también es necesario utilizar la corona circular, para que permita la lectura de pulsos procedentes del encoder óptico, siendo aconsejable volver a usar el soporte para que el encoder esté lo más estable posible y a una altura del eje del motor óptima para que la rueda implementada corte adecuadamente su haz de luz.

Como sucede anteriormente, dos son los archivos empleados para realizar este apartado, uno que se compila directamente en la placa de *Arduino* y otro que se visualiza desde el PC.

7.2.1. Control de velocidad Arduino

El programa para el control de velocidad que se transfiere a *Arduino* se trata de un lazo cerrado que permite comparar la entrada con la salida para poder tratar el error y actuar dependiendo de lo que se necesite. Este archivo, como en el caso de la Estática, se transfiere a *Arduino* con *Deploy to Hardware*.

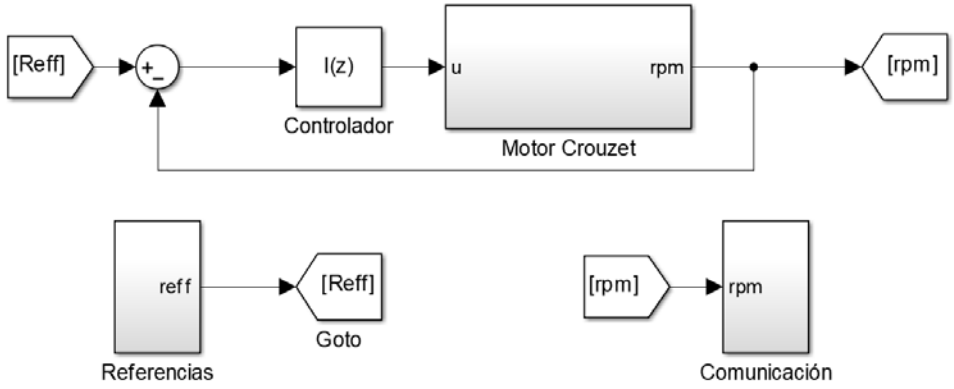


Imagen 22. Lazo de control de velocidad Arduino

Este lazo de control tiene englobado al motor *Crouzet*, encargado de comportarse de una u otra manera.

Como se puede apreciar en la imagen, el control de velocidad del motor de la servoválvula consiste en un lazo cerrado, donde es posible comparar la salida con la entrada y realizar los cálculos y correcciones necesarias para que la nueva entrada pase a ser la salida.

El lazo abierto no corrige el error existente entre la entrada y la salida, en ese momento, y como ocurre con el estudio de la estática, se introduce un valor al sistema y, tras procesarlo, se obtiene la salida sin compararla con la entrada. Mientras que en lazo

cerrado sí, permitiendo un control más preciso del sistema y obteniendo valores muy cercanos a los deseados.

El subsistema que hace la labor de actuador y planta del lazo cerrado es el *Motor Crouzet*, el cual tiene en su interior el envío de la acción de control del motor, que consiste en la función del *Driver L298N* explicada en el apartado de la Estática y que se encarga de enviar al driver los parámetros necesarios para controlar el sentido de giro del motor y su valor de velocidad.

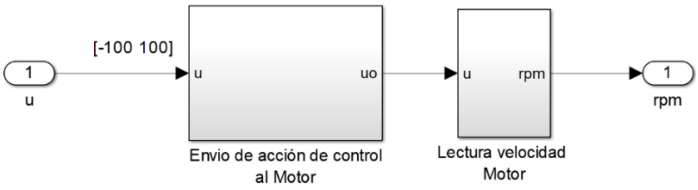


Imagen 23. Subsistema Motor Crouzet 82861010

Como se ha comentado, dentro del *Envío de acción de control* presenta la configuración de los parámetros para el envío de órdenes al *driver L298N*.

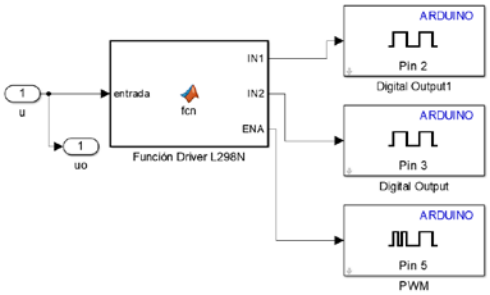


Imagen 24. Envío de acción de control, función Driver L298N

En el otro subsistema existente está la lectura de velocidad del motor, la cual engloba las funciones llamadas *Encoder* y *Filtro* necesarias para tratar la señal y transformarla en valores de velocidad.

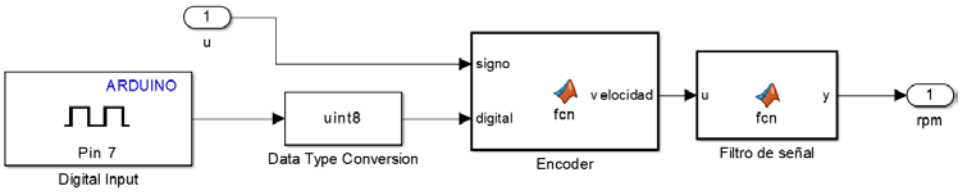


Imagen 25. Lectura de velocidad del Motor.

Este apartado es el mismo que se utiliza en el estudio de la *Estática* del motor. Leyendo del encoder óptico los pulsos para procesarlos y obtener la velocidad.

El siguiente bloque del lazo cerrado es el controlador del sistema, el cual si se quiere implementar en la vida real es necesario hacerlo en tiempo discreto, de ahí el motivo por el que se introduce un controlador discreto (dominio *z).* Como se explica en el apartado del *Diseño del controlador* se opta por implementar un controlador integral al sistema, el cual permite eliminar el error existente a la entrada de la acción de control y modificando su ganancia se consigue una respuesta lenta en lazo cerrado más rápida o más lenta.

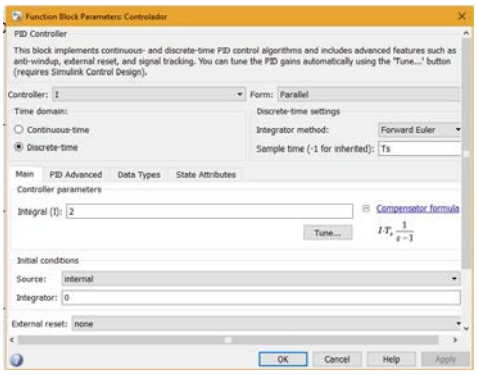


Imagen 26. Controlador implementado en velocidad

Los valores de entrada de velocidad están agrupados en el subsistema de *Referencias*, en este apartado se probó introduciendo diferentes entradas manualmente, obedeciendo a un potenciómetro o leyendo diferentes entradas analógicas que proporcionan el valor de señal deseado en el lazo cerrado de control.

El otro subsistema es el de *Comunicación* con el archivo de *Simulink* con el que se consigue la visualización de resultados de la velocidad en revoluciones por minuto (rpm), como sucede en *Estática*, la transmisión de datos se realiza a través del puerto serie.

Imagen 27. Recepción y transmisión de datos en el control de velocidad

7.2.2. Control de velocidad simulink

Los valores de velocidad en revoluciones por minuto se dividen entre 4 antes de ser convertidos a números de 8 bits enteros sin signo en la etapa de comunicación del archivo transferido a *Arduino*. Esto se hace para obtener una mayor precisión en los valores transmitidos, ya que se consigue una variación de valores casa 0.25 unidades en vez de cada unidad, como sucedería si se convirtiesen los valores obtenidos directamente en *uint8*.

Como en el archivo perteneciente al funcionamiento del *Arduino* se divide entre 4, en el *script* de lectura y visualización de valores que se crea se deben multiplicar los valores obtenidos por 4 tras ser convertidos de *uint8* a cifras con coma flotante *double*. Así se consigue visualizar una respuesta óptima de la respuesta del motor.

Los datos obtenidos de respuesta permiten adaptar la ganancia del controlador integral introducido. La parametrización de dicho valor es explicada en el apartado *Diseño de Controladores*.

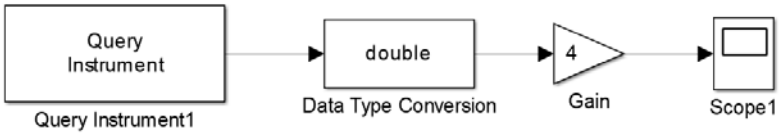


Imagen 28. Visualización de valores de velocidad en Simulink

7.3. Control de posición

7.3.1. Adecuación del potenciómetro

A la hora de comprobar el funcionamiento del potenciómetro se observó un problema, que era que una de sus patillas no funcionaba. Es decir, al usar un polímetro y medir la resistencia en cada una de las combinaciones posibles se observó que con la tercera patilla no se leía ninguna señal. Mientras que entre las dos primeras patillas sí que variaba la resistencia cuando se modificaba la señal del potenciómetro. Al no funcionar una de las patillas del potenciómetro, el instrumento no actuaba como divisor de tensión, por lo que se optó por implementar una resistencia en serie a una de las dos patillas activas para así poder conseguir el propósito descrito.

La resistencia introducida es de 1kΩ, añadiendo dicho valor al total que presentaba el potenciómetro en un principio. Inicialmente se podía asegurar una caída de tensión igual a la de alimentación del dispositivo, pero con la modificación siempre va a caer un mínimo de potencial entre los terminales donde está conectada la resistencia implementada. Se recomienda que la resistencia tenga un valor pequeño para que dicha caída no sea grande. En la imagen se muestra el conexionado del potenciómetro con la implementación de la resistencia en la segunda patilla (está cubierta por material termo retráctil).

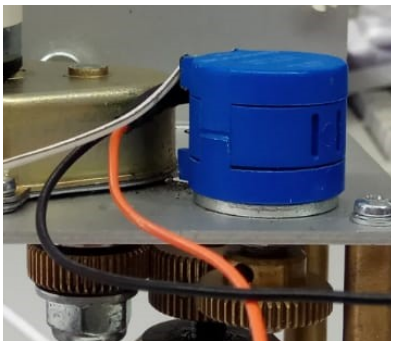


Imagen 29. Potenciómetro modificado

Como se ha comentado, siempre va a haber una caída de resistencia mínima en el circuito debido al valor constante de la resistencia, el cual será.

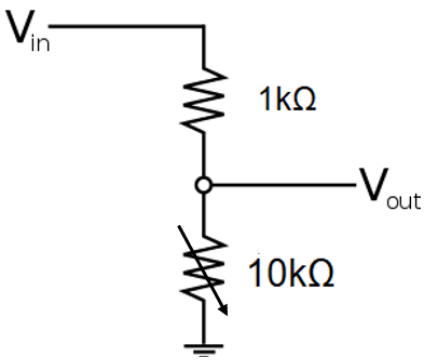


Imagen 30. Divisor de tensión

Si se aplican 5 voltios de tensión de entrada y el potenciómetro tiene un valor de cero se obtiene la salida mínima que presentará el divisor de tensión.

$$V_{out\ min} = \frac{1\ k\Omega}{10k\Omega + 1k\Omega} \cdot V_{in} = \frac{1\ k\Omega}{11\ k\Omega} \cdot 5V = 0.45\ V \qquad (7.2)$$

Experimentalmente se comprueba que, en efecto, la caída de tensión mínima pertenece a ese valor.

Se observó que el comportamiento del potenciómetro no es lineal, es decir, la relación entre posición y voltaje o resistencia a la salida no sigue un orden lineal. Variando el valor de resistencia del potenciómetro a la salida se puede comprobar que su comportamiento es logarítmico, pero se quiere que su valor siga un orden lineal para que el control de posición obedezca como se desea. Para ello se hacen diferentes medidas y pruebas para determinar las características de la ecuación logarítmica que lo representa.

Para determinarla se optó por medir el valor de tensión en cada paso por vuelta.

Ángulo girado/º	Número de vueltas	Valor tensión / V
36	0,1	0,2745
360	1	2,5490
720	2	3,3725
1080	3	3,7647
1440	4	4,0000
1800	5	4,1765
2160	6	4,2941
2520	7	4,3725
2880	8	4,4510
3240	9	4,4902
3600	10	4,5294

Tabla 3. Relación entre tensión y número de vueltas del potenciómetro usado

Como se observa en la tabla, y sabiendo el comportamiento logarítmico, se decide no recoger en la tabla el valor de tensión inicial ya que la función logarítmica no pasa por cero debido a sus características. Representando los valores se obtiene.

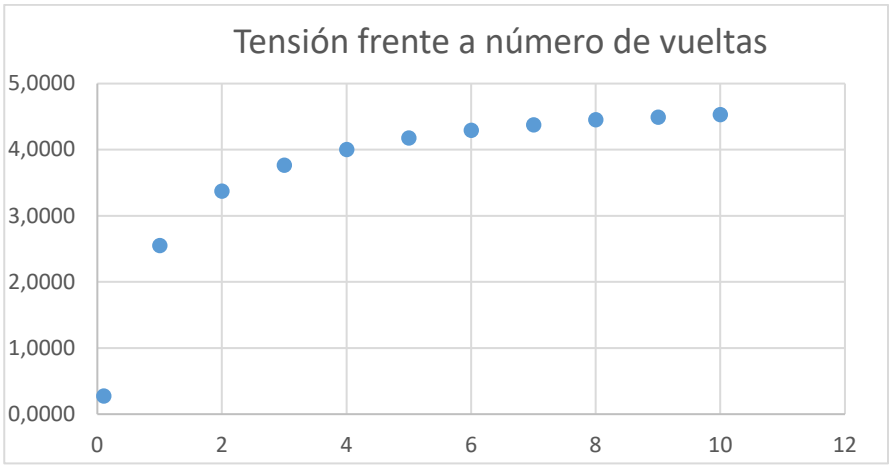


Imagen 31. Representación de la tensión frente al número de vueltas

En este caso se quiere estudiar el número de vueltas frente al ángulo girado por el potenciómetro, por lo que basta con multiplicar el número de vueltas por 360º que tiene cada una de ellas y en la representación se escogen todos los valores excepto el inicial para conseguir mayor precisión en la estimación de la línea de tendencia del sistema.

Se vuelve a representar la gráfica, sin el primer valor, y se calcula la línea de tendencia logarítmica del sistema.

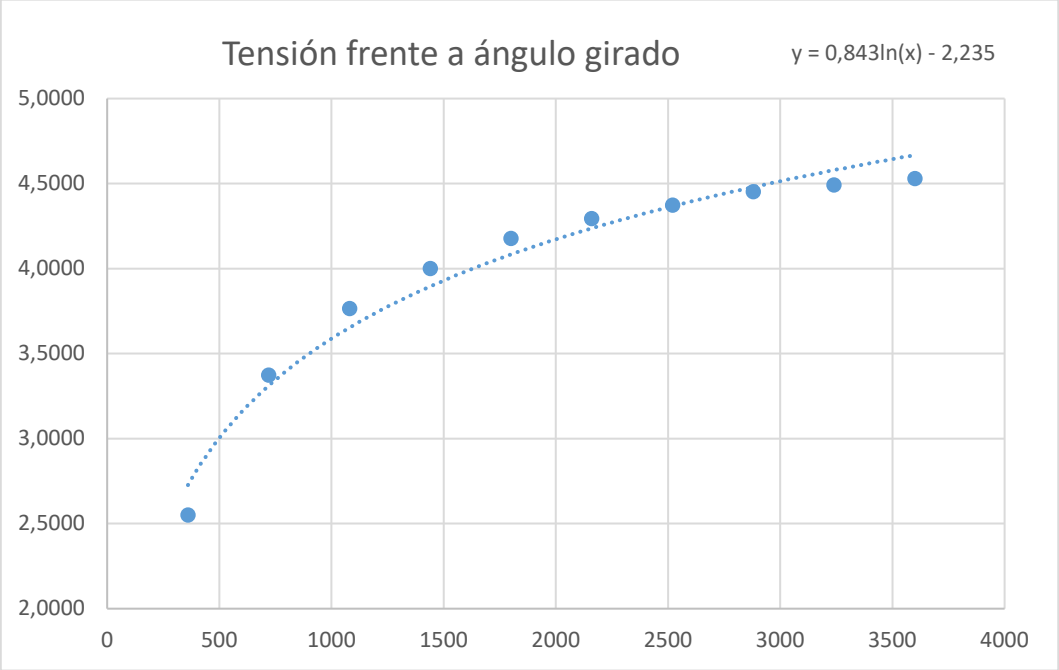


Imagen 32. Relación entre tensión y ángulo girado

De esta forma se calcula la línea de tendencia que describe el comportamiento del potenciómetro, la cual se usa en el control de posición para que, mediante programación, se consiga que una variación en la posición del potenciómetro de manera lineal.

7.3.2. Posición Arduino

El control de posición implementado en el conjunto de la servoválvula es el mostrado en la imagen.

La entrada es una señal analógica que va a determinar el valor de tensión. El cual se convierte en grados para tener la posición de referencia, a la que se desea llegar. El controlador empleado es de ganancia unidad, se explica en el apartado de *Diseño de controladores.*

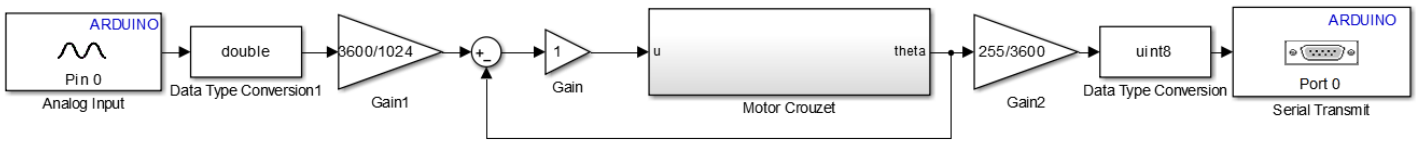


Imagen 33. Lazo de control de posición Arduino

El actuador, es decir, el *Motor Crouzet,* es un subsistema compuesto por dos partes, en primer lugar, está el *Envío de acción de control al motor,* que envía los datos correspondientes al *Driver L298N* para determinar el comportamiento del motor (es la misma función que en el lazo de control de velocidad). Y en segundo lugar está el subsistema perteneciente al potenciómetro cuyo movimiento es solidario al motor e indica la posición en la que está el actuador para que así el lazo de control pueda corregirla y obtener el valor deseado.

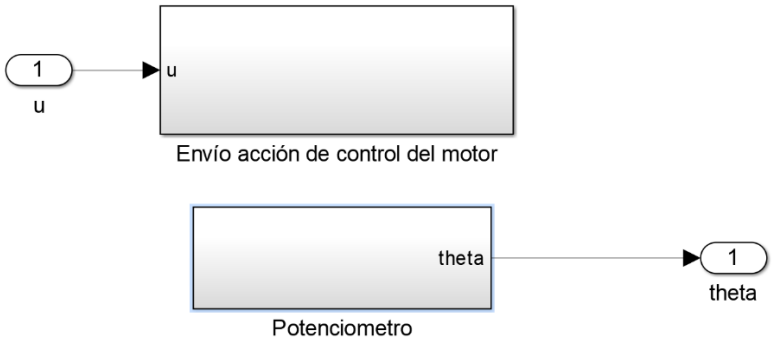


Imagen 34. Actuador Motor Crouzet Simulink

El subsistema referente a la lectura de valores del potenciómetro solidario al movimiento del motor es el mostrado a continuación.

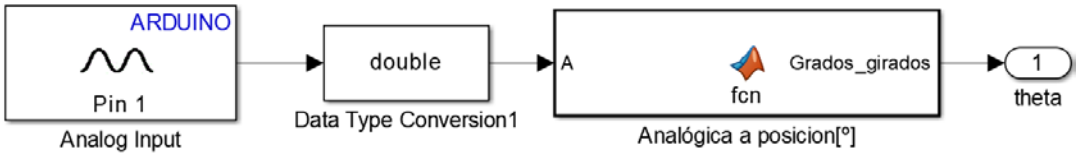


Imagen 35. Subsistema del potenciómetro del actuador en Simulink

En este subsistema se ha trabajado con la función logarítmica obtenida del comportamiento del potenciómetro real para conseguir que este dispositivo funcione de una manera lineal y como se desea.

```
function Grados_girados  = fcn(A)    % la entrada es la señal analógica del sistema
V=5-A*5/1024;                        % se realiza la conversion de analógica a tensión
Grados_girados=3600-exp((V+2.235)/0.843); %valor de referencia de la posición del motor
```

Pero el lazo de control implementado carece de límites tanto superior como inferior, por lo que al probarlo da problemas en dichas magnitudes, sobre todo en el límite inferior, en el cual se desacoplan el sistema de engranajes, es decir, la rueda dentada de la servoválvula de desacopla a la del potenciómetro de referencia de posición, perdiéndose así la medida y el control de posición del conjunto, por lo que se decide crear una función que delimite la entrada al sistema, asignándole un rango de valores entre los que puede estar.

Los valores impuestos son los observados experimentalmente, llevando al límite al engranaje y visualizando cuando está completamente cerrada o abierta la válvula. De manera que el código introducido está compuesto por una serie de condiciones.

De esta forma el sistema se mantiene en el valor mínimo indicado cuando la entrada sea menor o igual a ese mínimo, pasando lo mismo con el límite superior. De esta forma se crea una mayor seguridad en el sistema, consiguiendo así que no se desacoplen los engranajes propios del movimiento del conjunto.

```
function sistema = fcn(entrada)
%#codegen

if entrada <= 550;
    sistema=550;            % se le asigna el valor mínimo al sitema

else if entrada >= 3479;
   sistema=3479;            % se le asigna el valor máximo al sistema
    else
   sistema=entrada;         % si la entrada está entre ese rango de valores, la salida
es igual a ella
    end
end
end
```

Por lo que el sistema queda de la siguiente forma.

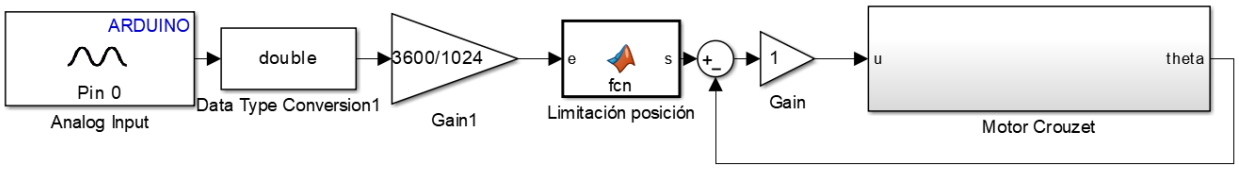


Imagen 36. Lazo de posición con la limitación introducida

7.3.3. Posición Simulink

Para visualizar la posición a la que va el sistema se puede utilizar un esquema básico de *Simulink*, el cual se muestra a continuación.

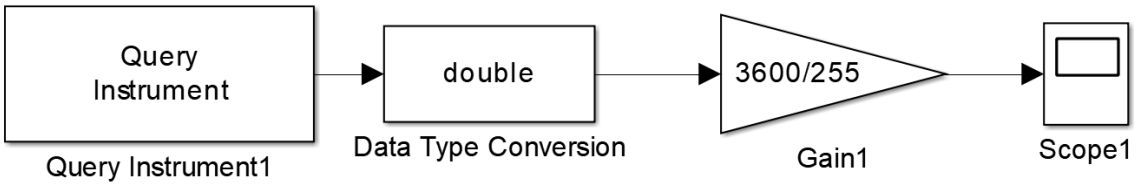


Imagen 37. Visualización de valores de posición en Simulink

8. Obtención de las plantas de velocidad y posición

8.1. Cálculo experimental de la planta para control de velocidad

En este apartado se obtiene la función de transferencia que describe el comportamiento dinámico del motor de corriente continua *Crouzet 82861010* cuando éste es excitado con una señal.

Los archivos de *Matlab & Simulink* utilizados en este caso son los de la *Estática* y la señal que se usa para identificar el motor es una *escalera*. Con esta señal se alimenta el motor y se obtienen los datos de la respuesta para poder procesarlos y adquirir una expresión numérica que relacione el valor *PWM* que se aplica al motor y a la velocidad angular obtenida.

Este apartado se centra en estudiar la estática y la dinámica del motor, para determinarlas y llegar a una conclusión se hicieron varias pruebas, en las cuales se observó de manera gráfica su respuesta en velocidad y, a partir del análisis experimental de determinaron los parámetros que definen su respuesta transitoria. La *escalera* de entrada se genera en un *"script"* de *Matlab* y sus valores representan el porcentaje de potencia que se le aplica al motor en *PWM*.

```
%% Identificación de la estática del motor

V_min = -100;   % Tensión mínima
V_max = 100;    % Tensión máxima
AV = 20;        % Incremento de voltaje entre escalón y escalón.
t_escalon = 5;  % Tiempo entre escalón y escalón de la excitación.
Ts=0.01;

n_escalones = (V_max-V_min)/AV + 1;
tf = n_escalones * t_escalon;
t = 0:Ts:tf-Ts;

for i=1:n_escalones
u(1+(i-1)*t_escalon/Ts:i*t_escalon/Ts) = V_min + (i-1)*AV;
end
h1=plot(t, u);                          % graficamos los valores obtenidos
title('Escalera de valores de entrada');% introducimos el título principal
xlabel('tiempo [s]');                   % introducimos el título del eje X
ylabel(' Potencia [%]');                % introducimos el título del eje Y
axis([0 55 -100 100]);                  % configuramos los límites de cada eje
grid on;                                % malla

% Guardamos los datos en una variable llamada "Escalera"
Escalera.time = t';
Escalera.signals.values = u';
```

La escalera descrita está formada por escalones de 20 unidades con una duración de 5 segundos cada uno *(Imagen 38).*

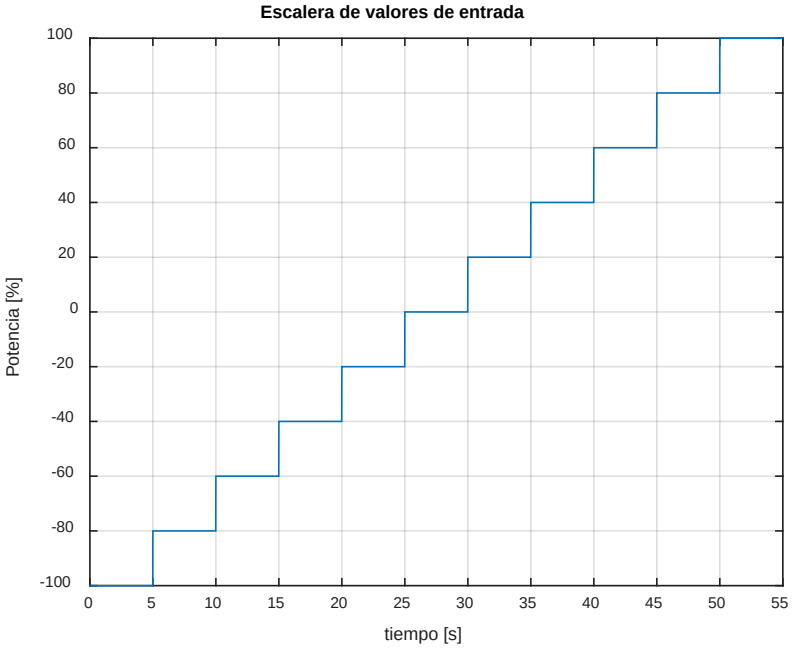


Imagen 38. Gráfica de la escalera de entrada al motor DC

La velocidad de la respuesta del motor de corriente continua ante la escalera de entrada es mostrada en grados por segundo.

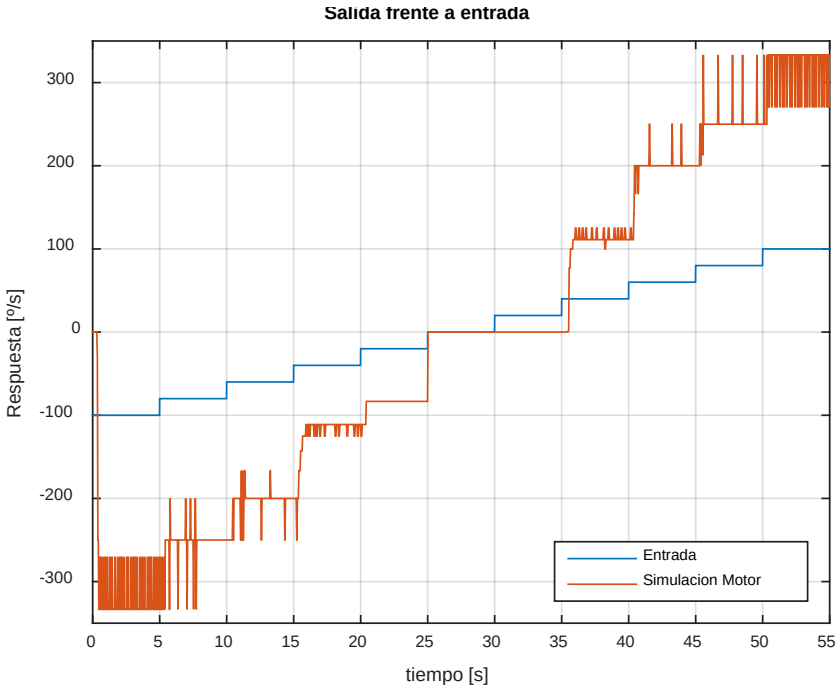


Imagen 39. Gráfica de la respuesta a la escalera del motor DC

8.1.1. Primera prueba

Utilizando la herramienta *Ident* de *Matlab* y usando los datos obtenidos en el experimento se llega a una primera planta que puede representar el comportamiento del sistema real. La planta consta de una ganancia y un único polo, es decir, se trata de un sistema de primer orden.

Como se ha comentado, se utilizó la herramienta *Ident,* para tomar esta primera posible planta del sistema. Lo primero que se hizo fue guardar la respuesta del motor en una variable llamada *Step*, y se preparó el código para escoger un tramo de dicha respuesta que contenga poco ruido y que muestre el comportamiento del motor.

```
%% Tramo tomado para calcular la planta en º/s
t1=23;                                          % primer tiempo
t2=28;                                          % segundo tiempo del tramo
u_dinamica=Step.signals(1).values(t1/Ts:t2/Ts); % entrada al sistema
y_dinamica=Step.signals(3).values(t1/Ts:t2/Ts); % respuesta a la excitación
```

Se abre la herramienta *Ident* para estudiar la respuesta al escalón escogido.

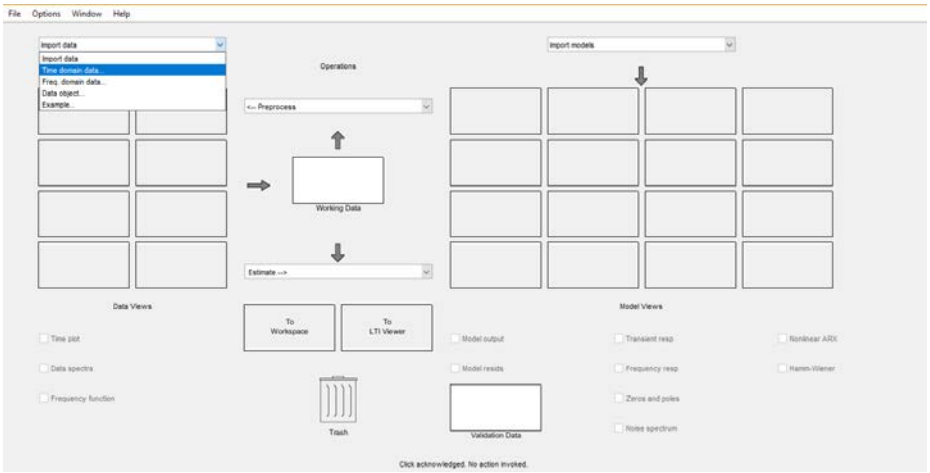


Imagen 40. Visualización de la función Ident

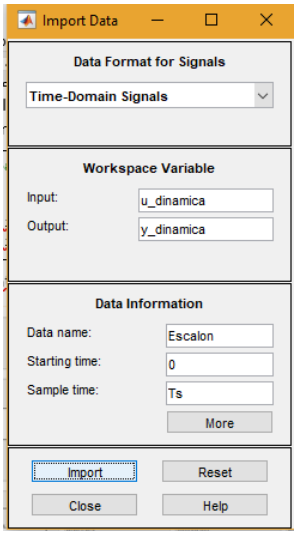


Imagen 41. Implementación del escalón a analizar

Una vez elegido el escalón se analiza, para ello se pincha en la pestaña *Estimate* y se selecciona la opción *Process Models.*

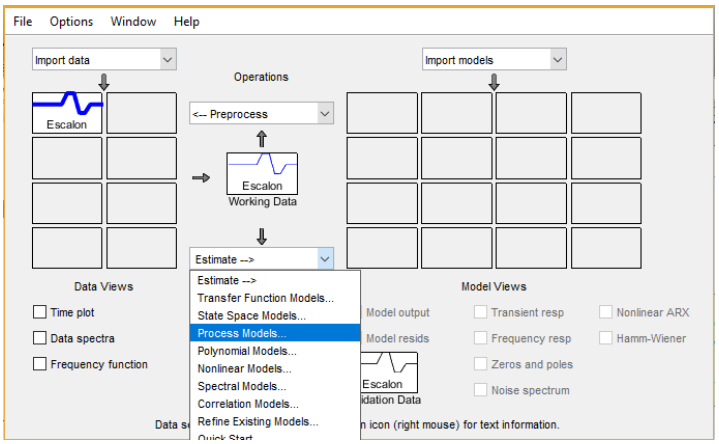


Imagen 42. Selección de Process Models para obtención de planta

A continuación, se escoge como se quiere que sea la respuesta, es decir, los polos y los ceros que se desean, así como el retardo o demás parámetros que se pueden introducir. Lo más normal es que un motor de corriente continua esté caracterizado por una planta de primer orden, por lo que se escoge un sistema que contenga ganancia y un polo en el eje real.

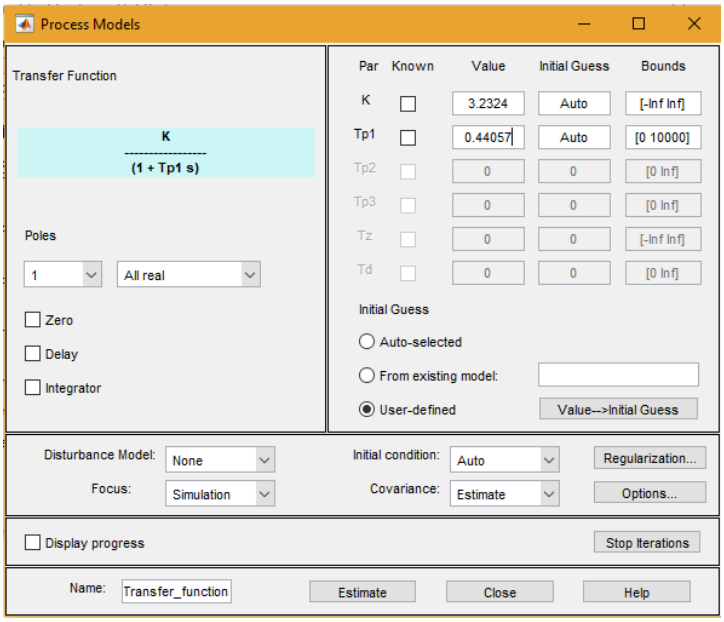


Imagen 43. Selección de parámetros para la obtención de la planta

La planta obtenida es la siguiente (8.1).

$$Planta(s) = \frac{3.2324}{0.44057s + 1} \tag{8.1}$$

Para observar la adecuación de la planta calculada con la respuesta real del motor de corriente continua se selecciona *Model Output.* Como se puede apreciar, la adecuación es de un 64,82%.

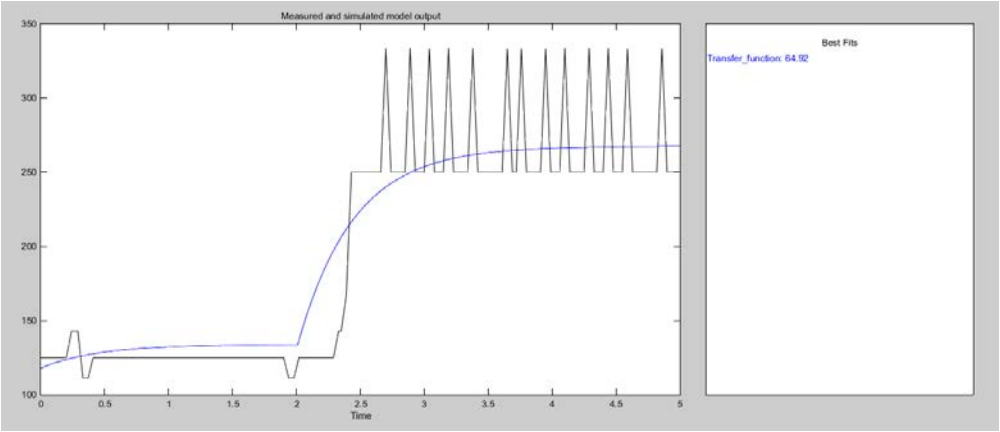


Imagen 44. Adecuación de la planta obtenida a la respuesta real

Como se puede observar, la planta obtenida presenta un retraso respecto al escalón analizado, es decir, que es más lenta que la respuesta del motor en la vida real.

Para representar la respuesta de la planta a la escalera de valores de entrada se genera un *Script* en *Simulink* cuya entrada sea la escalera, luego se usa el bloque *LookUp Table* para simular la ganancia del sistema, introduciendo únicamente en el bloque *transfer function* la dinámica, es decir, el polo de la planta obtenida.

```
%% Grafica Look-Up Table (º/s)
urp=V_min:AV:V_max;

for i=1:n_escalones
t3=3.5+(i-1)*5;
t4=5+(i-1)*5;
datos=OtraPrueba2.signals(3).values(t3/Ts:t4/Ts);
media=sum(datos)/length(datos);
yrp(i)=media;
end
plot(urp,yrp);
title('Modelo estático');
xlabel('t [s]');
ylabel(' Salida [º/s]');
axis([0 55 -350 350]); grid on; grid minor;
```

Los valores que introducir en el bloque de Simulink *LookUp Table* que representan la ganancia del sistema son:

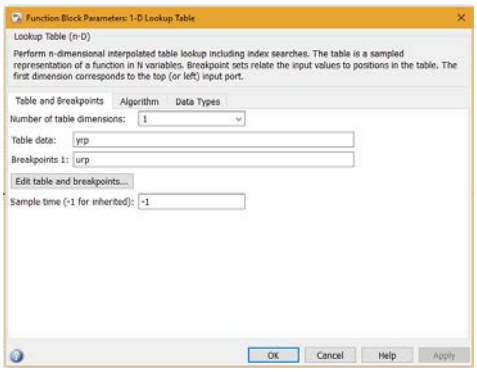


Imagen 45. Configuración de los parámetros de la LookUp Table (Simulink)

Del código de *Matlab* descrito arriba se obtiene el siguiente modelo estático del motor, expresando la velocidad en grados por segundo.

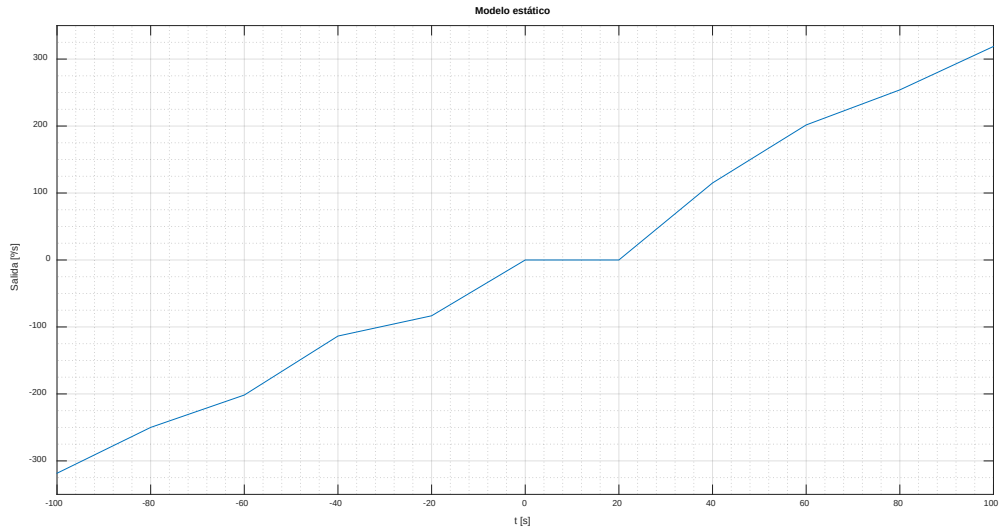


Imagen 46. Gráfica del modelo estático de la respuesta real

Se simula en lazo abierto la respuesta de la planta del motor a la escalera, como se ha explicado, la ganancia de la planta se representa con la *LookUp Table*.

Imagen 47. Lazo abierto con la simulación de la planta obtenida

Para configurar el *scope* se pincha en él y en la pestaña *general* se indica que se quieren dos *axes,* luego, en *History* se deselecciona el límite de puntos que está puesto por defecto y se clicka en *save data to workspace*, para guardar los datos simulados y obtenidos, que van a tener un formato de *structure with time.*

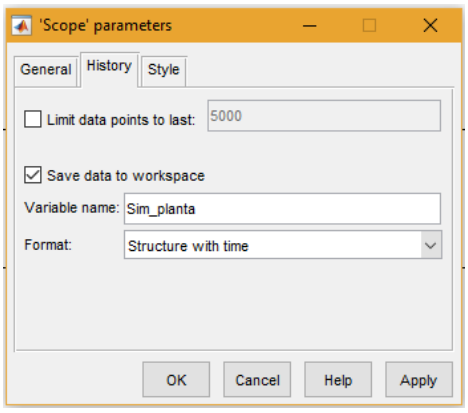


Imagen 48. Configuración del Scope

Ahora, para graficar la respuesta de la planta obtenida y representarla frente al comportamiento real del motor se realiza el siguiente código.

```
%% Representación del comportamiento de la planta de primer orden
obtenida con la función ident
figure
h2=plot(OtraPrueba2.time,OtraPrueba2.signals(3).values,
Sim_planta.time, Sim_planta.signals(2).values);
title('Representacion de respuestas obtenidas');
xlabel('t [s]');
ylabel(' Salida [°/s]');
legend(h2,   'Simulacion   Motor',   'Planta   obtenida','Location',
'southeast');
axis([0 55 -350 350]); grid on; grid minor;
set(h2,{'LineWidth'},{1;1.5});
```

Así se puede visualizar y comparar el comportamiento de la respuesta real del motor y la de la planta simulada para poder obtener conclusiones.

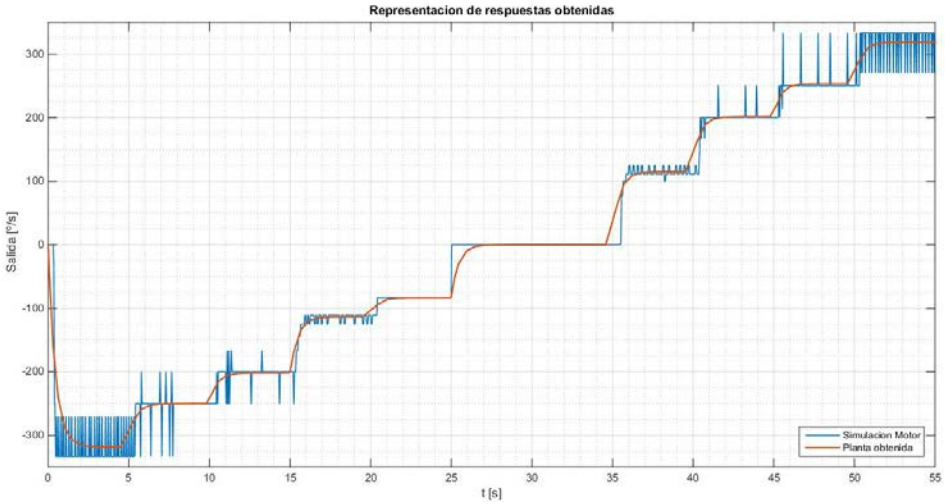


Imagen 49. Gráfica de respuesta real frente a planta simulada

Comparando y observando este comportamiento se llega a la conclusión de que la planta característica de nuestro sistema es de tipo ganancia pura, es decir, no tiene dinámica. Esto es debido a que la ecuación del sistema obtenida con la función *ident* es más lenta y que, como se puede apreciar en la respuesta del motor, su comportamiento es parecido a un escalón.

8.1.2. Segunda prueba

Tras analizar los primeros cálculos realizados se concluye entonces que la planta es una ganancia pura, la cual se procede a calcular ahora tomando los datos de los valores de la *datasheet.* En ella está descrito que la velocidad máxima del motor que puede alcanzar al aplicarle el máximo valor de entrada (12 voltios) es de 4300 revoluciones por minuto, también aparece que el ratio de la reductora de 80, es decir, que el valor máximo de velocidad de salida es:

$$Output\ speed = Vel.máxima\ Motor\ \cdot Ratio\ reductora = 4300\ rpm \cdot \frac{1}{80} = 54\ rpm \quad (8.2)$$

Se confirma el valor observando de nuevo la *datasheet,* que indica que el valor máximo de velocidad de salida del eje, es decir, tras pasar por la reductora, es de 54 revoluciones por minuto.

Por lo que se puede calcular la nueva planta del sistema para el control de velocidad, la cual se calcula tomando los datos de tensión máxima de entrada de 12 voltios y velocidad máxima de salida del eje de 54 revoluciones por minuto:

$$P_2(s) = \frac{54\ rpm}{12\ Voltios} = 4.5 \tag{8.3}$$

8.2. Obtención de la planta para control de posición

Como se ha explicado en el control de velocidad, el ratio de la reductora que lleva incorporada el motor tiene un factor $\frac{1}{80}$. Es decir, la velocidad del motor queda dividida por el índice de la reductora antes de convertirse en la velocidad de salida. En el sistema el eje de salida ya tiene incorporada la reductora, por lo que la velocidad del motor ya habrá pasado por ella y el valor medible que se obtiene es el de salida.

Para obtener la posición a la salida del sistema en lazo cerrado se integra el valor de velocidad procesado, en el dominio "*s"* integrar significa dividir por "*s"*. Es decir, la planta para obtener la posición del sistema de la servoválvula es un integrador. La ganancia de este integrador es de una unidad porque la reductora ya está acoplada al eje de salida del motor, es decir, la velocidad ya ha pasado por dicho dispositivo y no hace falta volver a dividirlo por su ratio para obtener la posición real en la que está el motor. Llegando a la conclusión que la planta para realizar el estudio de la posición es:

$$P_1(s) = \frac{1}{s} \tag{8.4}$$

9. Diseño de controladores de velocidad y posición

9.1. Introducción

Cuando un lazo de control va a ser implementado aparecen una serie de criterios a seleccionar para ser controlados: que el bucle sea estable, que los efectos de las perturbaciones se minimicen, que se obtengan respuestas rápidas y suaves, que el sistema sea robusto, es decir, poco sensible a los cambios en las condiciones de proceso o debido a los errores.

En principio cualquier propiedad puede considerarse para seleccionar la respuesta del sistema, por ejemplo: rebasamiento máximo ("*overshoot"),* tiempo de decaimiento ("*rise time*"*,* hasta alcanzar el valor deseado por primera vez), tiempo de establecimiento ("*settling time"*, hasta quedar en ±2% del valor deseado), relación de decaimiento ("*decay ratio*"*,* se trata de la relación entre la altura del segundo y el primer pico), frecuencia de oscilación, etc.

Los controladores utilizados tradicionalmente en la industria son los controladores proporcional (P), proporcional integral (PI) y proporcional integral derivativo (PID). Estos controladores se visualizan y se explican mejor teniendo en cuenta el lazo de control que se muestra en la imagen.

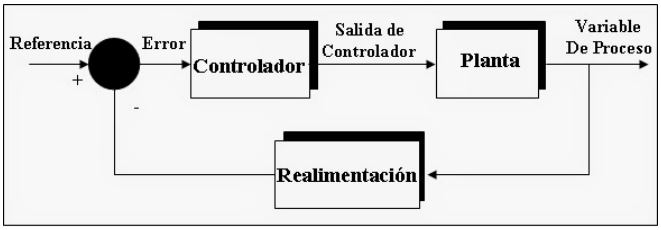


Imagen 50. Lazo cerrado de control

Si se utiliza el controlador proporcional, la salida entonces es proporcional al error, como se muestra en la siguiente ecuación.

$$u(s) = K_P \cdot e(s) \tag{9.1}$$

El controlador proporcional no modifica la forma del lugar de las raíces (condición de argumento). Además, un controlador P no elimina el error en estado estacionario (a entrada escalón) para plantas del tipo cero. El controlador PI presenta un polo en el origen, quedando así la función de transferencia del sistema como:

$$u(s) = K_P \cdot e(s) + \frac{1}{T_i s} \cdot e(s) \tag{9.2}$$

Donde T_i es la constante de tiempo integral y K_p la constante proporcional. El controlador PI sí que es capaz de eliminar el error en estado estacionario a la entrada del escalón para plantas de tipo cero (o superior).

El siguiente controlador mostrado es el PID (controlador proporcional integral derivativo), se trata del más utilizado y común. Puede representarse por la siguiente ecuación.

$$u(s) = \left(K_P + \frac{1}{T_i s} + T_d s\right) \cdot e(s) \tag{9.3}$$

Como se puede observar, este controlador tiene tres grados de libertad K_p, T_i y T_d, lo que permite colocar dos ceros en cualquier lugar dentro del plano complejo, más un polo en el origen.

La función de transferencia del PID puede escribirse también como:

$$u(s) = K_P \cdot \left(\frac{s^2 + as + b}{s}\right) \tag{9.4}$$

En esta expresión a y b son ajustados mediante la variación de la constante del tiempo integral y derivativo. La principal ventaja que presentan los PID es que pueden ser utilizados para compensar plantas como polos complejos poco amortiguados.

9.2. Controlador usado para el ajuste de velocidad

Obtenida la planta en el control de velocidad, se procede a calcular el controlador más adecuado. Como la planta es de ganancia pura y se desea que el controlador elimine los errores para que la respuesta del sistema sea lo más precisa posible, se opta por escoger un integrador.

En este apartado también se realizan varias pruebas para elegir el valor más óptimo de la ganancia del integrador. Con los diferentes valores se observa cómo y cuánto tarda el sistema en reaccionar ante las diferentes entradas.

9.2.1. Primera prueba

En primer lugar, se prueba un controlador de respuesta lenta, es decir, con poca ganancia.

$$C_2(s) = \frac{0.2}{s} \tag{9.5}$$

La función de transferencia es la relación entre salidas y entradas del sistema, en este caso la salida del sistema es *Y(s)* y la entrada corresponde con *R(s).* De forma que queda representada por.

$$Función\ de\ transferencia = \frac{Y(s)}{R(s)} \tag{9.6}$$

La función de transferencia del lazo cerrado de un sistema formado por el controlador y la planta es:

$$\frac{Y(s)}{R(s)} = \frac{C_2(s)\cdot P_2(s)}{1+C_2(s)\cdot P_2(s)} = \frac{\frac{numC_2(s)}{denC_2(s)}\frac{numP_2(s)}{denP_2(s)}}{1+\frac{numC_2(s)\cdot numP_2(s)}{denC_2(s)\cdot denP_2(s)}} \tag{9.7}$$

$$\frac{Y(s)}{R(s)} = \frac{numC_2(s)\cdot numP_2(s)}{denC_2(s)\cdot denP_2(s)+numC_2(s)\cdot numP_2(s)} \tag{9.8}$$

Sustituyendo en la última ecuación los valores que se tienen del controlador y la planta del motor *Crouzet 82861010* se tiene que:

$$\frac{Y(s)}{R(s)} = \frac{numC_2(s)\cdot numP_2(s)}{denC_2(s)\cdot denP_2(s)+numC_2(s)\cdot numP_2(s)} = \frac{0.2\cdot 4.5}{s\cdot 1+0.2\cdot 4.5} \tag{9.9}$$

$$\frac{Y(s)}{R(s)} = \frac{0.9}{s+0.9} \rightarrow 1er\ orden \rightarrow \frac{Y(s)}{R(s)} = \frac{K}{\tau s+1} \tag{9.10}$$

Al ser una función de transferencia de primer orden, el tiempo de establecimiento del sistema en la banda del dos por ciento es 4τ. Pero antes se debe calcular el valor de τ.

$$\frac{s}{0.9} + \frac{0.9}{0.9} = \tau s + 1 \rightarrow 1.11s + 1 = \tau s + 1 \rightarrow \tau = 1.11 \tag{9.11}$$

Una vez conocido el valor de τ se procede a calcular el tiempo de establecimiento del sistema añadiendo el primer controlador integral aplicado.

$$ts_{2\%} = 4 \cdot \tau = 4 \cdot \frac{1}{0.9} = 4.4\ segundos \tag{9.12}$$

Tras los cálculos realizados se procede a justificar sus valores en *Matlab*.

```
s=tf('s');          % se crea el entorno en s
P=54/12;            % se introduce la planta del sistema
C=0.2/s;            % se introduce el primer controlador propuesto
H=feedback(C*P,1);  % se genera y se guarda en H el lazo cerrado con realimentación unitaria
step(H)             % se simula la respuesta al escalón del lazo cerrado
```

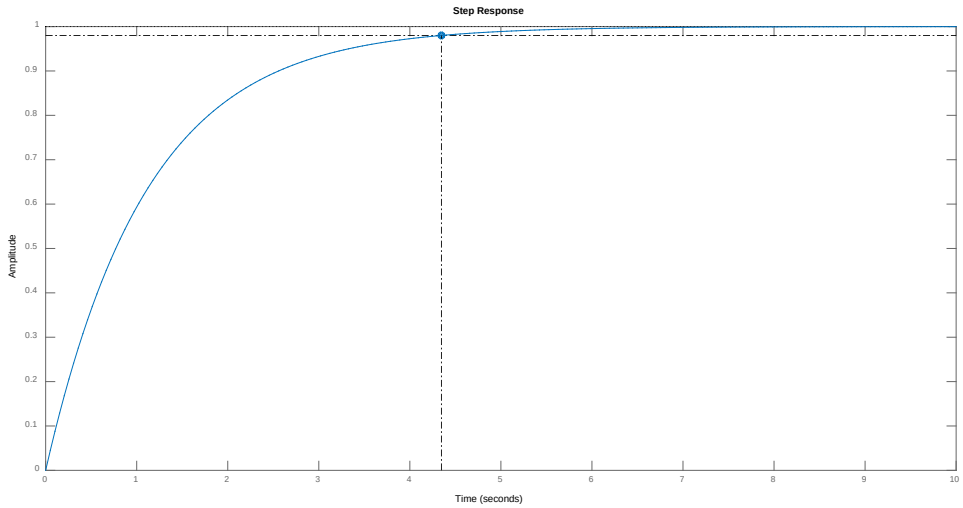


Imagen 51. Respuesta al escalón del primer controlador probado

Como se puede comprobar, el tiempo de establecimiento del sistema está situado en 4.4 segundos. Es decir, el sistema tarda alrededor de 4.4 segundos en volver a asentarse tras una variación en su entrada.

9.2.2. Segunda prueba

Se ha comentado antes que esta primera prueba de controlador iba a tener una respuesta lenta, ya que el controlador es poco agresivo debido a su poca ganancia (0.2). Esta prueba era una primera toma de contacto para ver que el sistema real se comporta igual que en la simulación, tal y como se desea.

Ahora se va a proceder a introducir una ganancia diez veces más agresiva, de modo que se reduzca en gran manera el tiempo de respuesta del sistema.

$$C(s) = \frac{2}{s} \tag{9.13}$$

$$\frac{Y(s)}{R(s)} = \frac{C_2(s)\cdot P_2(s)}{1+C_2(s)\cdot P_2(s)} = \frac{2\cdot 4.5}{s\cdot 1+2\cdot 4.5} \rightarrow \frac{Y(s)}{R(s)} = \frac{9}{s+9} \rightarrow 1^{er} orden \rightarrow \frac{Y(s)}{R(s)} = \frac{K}{\tau s+1} \tag{9.14}$$

El tiempo de establecimiento en la banda del dos por ciento es:

$$ts_{2\%} = 4 \cdot \tau = 4 \cdot \frac{1}{9} = 0.44\ segundos \tag{9.15}$$

```
s=tf('s');          % se crea el entorno en s
P=54/12;            % se introduce la planta del sistema
C=2/s;            % se introduce el primer controlador propuesto
H=feedback(C*P,1);  % se genera y se guarda en H el lazo cerrado con realimentación unitaria
step(H)             % se simula la respuesta al escalón del lazo cerrado
```

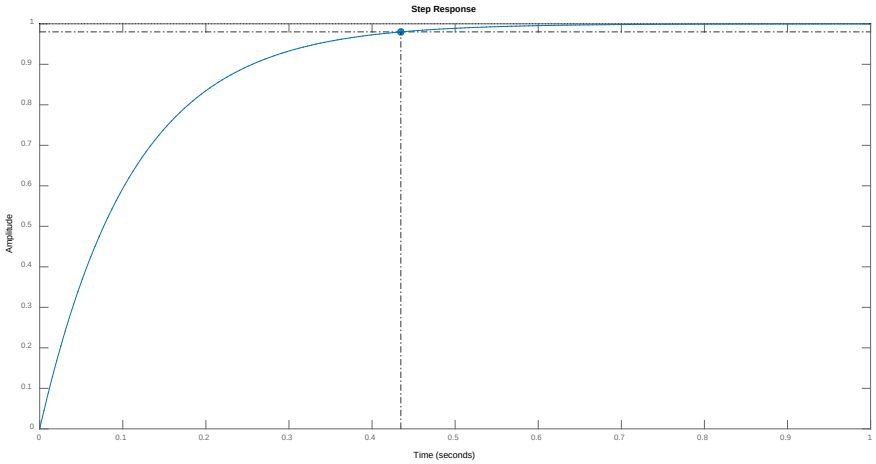


Imagen 52. Respuesta al escalón del segundo controlador empleado

Tal y como se puede apreciar, la ganancia del controlador es inversamente proporcional al tiempo de respuesta del sistema, es decir, cuanto más aumenta su valor menor va a ser el tiempo de establecimiento del sistema.

9.2.3. Implementación de la ley de control de velocidad en Simulink

En este apartado se muestran diferentes respuestas y comportamientos del control de velocidad del motor estudiado *Crouzet 82861010* para observar la diferencia existente entre introducir un controlador u otro en la vida real. En simulación se debe tener en cuenta que existe un retardo al enviar y procesar la señal debido a las comunicaciones PC-*Arduino* y *Arduino*-PC. Por este motivo es conveniente que todos los cálculos sean realizados en el microcontrolador *Atmel AVR*, así se conseguirá una reducción de tiempo de procesamiento de datos y lo que se perdería sería principalmente en la comunicación.

La siguiente imagen muestra la respuesta al escalón del control de velocidad implementándole el primer controlador descrito.

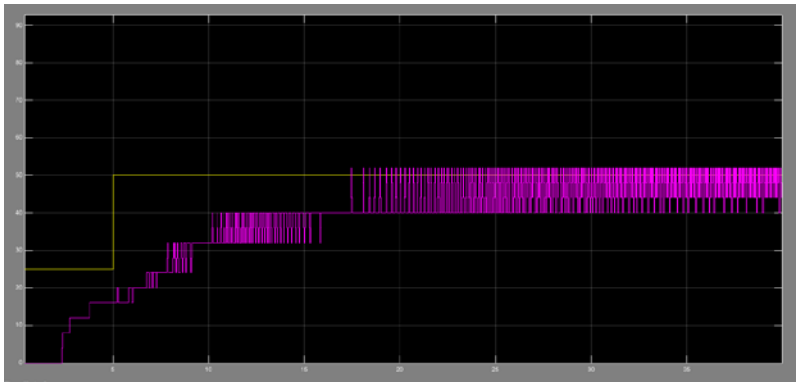


Imagen 53. Respuesta de velocidad con el primer controlador empleado

Tras observar esta respuesta se visualiza la siguiente, utilizando un controlador integral 10 veces mayor que el primero $C_V(s) = \frac{2}{s}$.

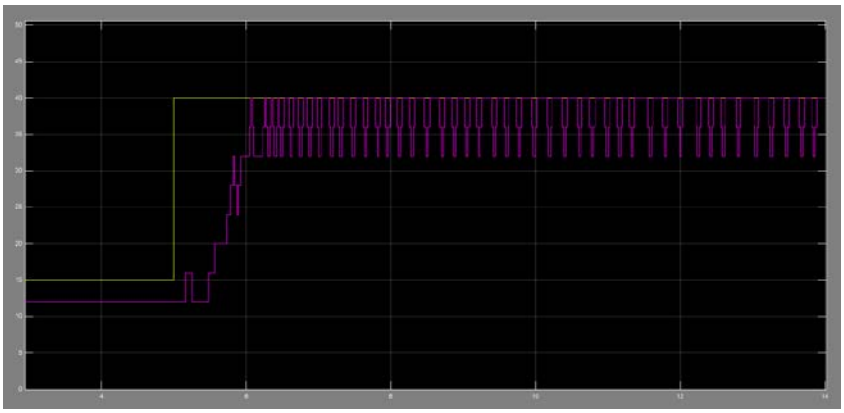


Imagen 54.Respuesta de velocidad con el segundo controlador empleado

Como se puede apreciar, la respuesta del primer controlador es mucho más lenta que el que finalmente se usa para realizar el control de velocidad del motor *Crouzet 82861010,* también se puede observar que el tiempo de respuesta es mayor al teórico en la simulación por la comunicación de datos.

9.3. Controlador usado para el ajuste de posición

Debido a que la planta de posición ya posee en su sistema un integrador (una *s* en el denominador) ya se elimina el error de posición que puede tener la respuesta en lazo cerrado respecto a la referencia, por lo que no es necesario introducir un controlador sofisticado, como puede ser un PI, un PD o un PID. Si no que bastaría, en principio, con un controlador de ganancia para ajustar la velocidad de respuesta del sistema.

9.3.1. Diseño y simulación de un controlador PID

Una vez cerrado el lazo de control de velocidad, se procede a cerrar el lazo de control de posición diseñando un controlador óptimo para conseguir una buena respuesta en simulación. El lazo de control en cascada queda de la siguiente forma.

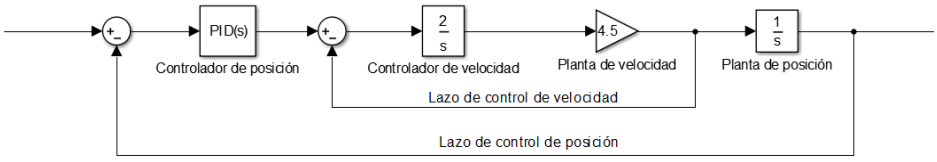


Imagen 55. Control en cascada del sistema

Como se ha observado en el apartado de la "*Segunda prueba"* para el control de velocidad, la función de transferencia del lazo de control de la propia velocidad queda de la siguiente manera.

$$\frac{Y(s)}{R(s)} = \frac{C_2(s)\cdot P_2(s)}{1+C_2(s)\cdot P_2(s)} = \frac{2\cdot 4.5}{s\cdot 1+2\cdot 4.5} \rightarrow \frac{Y(s)}{R(s)} = \frac{9}{s+9} \tag{9.16}$$

Una vez conocida la función de transferencia de velocidad $\left(\frac{Y_v(s)}{R_v(s)} = \frac{9}{s+9}\right)$ y sabiendo que la planta de posición consiste en un integrador debido a que la reductora del motor

está ya acoplada a su eje de salida. El lazo de control de posición es el siguiente, teniendo en la parte de la derecha la planta de posición una vez multiplicada la función de transferencia de velocidad con el integrador de posición. En este momento se puede diseñar un controlador PID industrial con las características que se deseen.

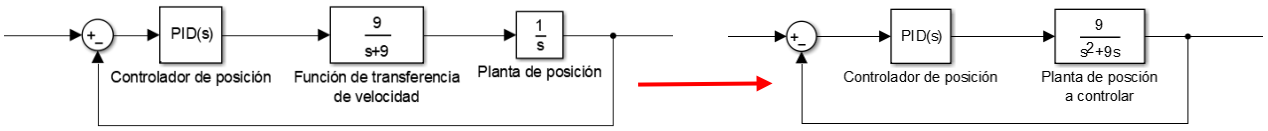


Imagen 56. Lazo cerrado teórico de control de posición

En primer lugar, se deben conocer algunas características del sistema sin controlar para determinar en qué se puede mejorar, para ello se estudia el lazo cerrado de control sin implementar ningún controlador. La función de transferencia queda:

$$\frac{Y_p(s)}{R_p(s)} = \frac{C(s)\cdot P(s)}{1+C(s)\cdot P(s)} = \frac{9}{s^2+9s+9} = \frac{K}{s^2+2\cdot\delta\cdot\omega_n\cdot s+\omega_n^2} \tag{9.17}$$

Tal y como se puede observar, la función de transferencia es de segundo orden, con lo cual su tiempo de establecimiento responde a la siguiente ecuación, donde δ es el amortiguamiento relativo y ω_n, la pulsación natural.

$$t_{s(2\%)} = \frac{4}{\delta\cdot\omega_n} = \frac{4}{4.5}s = 0.888\ s \tag{9.18}$$

Una vez conocidas las características iniciales del lazo de posición se diseña un controlador PID para que el sistema cumpla que su rebasamiento máximo sea del 4.3% y su tiempo de establecimiento, 0.4 segundos, es decir, se desea que el conjunto sea más rápido.

Primero se estudia si se pueden obtener los polos deseados sin modificar el lugar de las raíces.

$$RM(\%) = 100\cdot e^{\frac{-\delta\cdot\pi}{\sqrt{1-\delta^2}}} = 4.3\% \ \rightarrow\ \delta = \frac{1}{\sqrt{2}} = 0.707 \tag{9.19}$$

$$t_{s(2\%)} = 0.4\ s = \frac{4}{\delta\cdot\omega_n} \rightarrow \omega_n = \frac{4}{0.4\cdot\delta} = 10\cdot\sqrt{2} \tag{9.20}$$

Los polos deseados son:

$$Pd(s) = -\delta\cdot\omega_n \pm j\omega_n\cdot\sqrt{1-\delta^2} = -10 \pm j10 \tag{9.21}$$

Con estos datos se calcula la deficiencia angular.

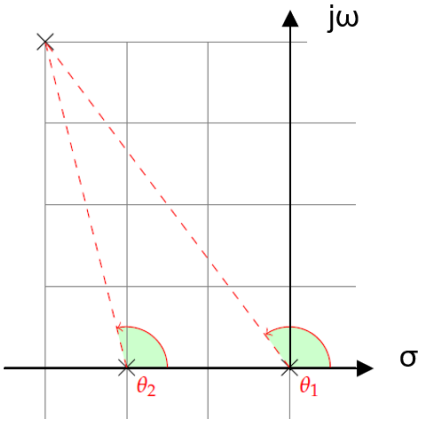


Imagen 57. Cálculo de los ángulos necesarios para obtener la deficiencia angular

$$|P(s)|_{s=Pd} + DA = (2 \cdot r + 1) \cdot 180° \quad (9.22)$$

$$\theta_1 = 180 - arctan\left(\frac{10}{10}\right) = 135° \quad (9.23)$$

$$\theta_2 = 180 - arctan\left(\frac{10}{1}\right) = 95.7106° \quad (9.24)$$

$$DA - \theta_1 - \theta_2 = (2 \cdot r + 1) \cdot 180° \rightarrow DA - 135° - 95.7106° = (2 \cdot r + 1) \cdot 180° \quad (9.25)$$

$$DA = 50.71° \quad (9.26)$$

El lugar de las raíces no pasa por los polos deseados, por lo tanto, se necesita emplear un controlador de adelanto de fase que mejore la dinámica del sistema. Se puede emplear un PD o una célula de adelanto de fase, en este caso se emplea un PD.

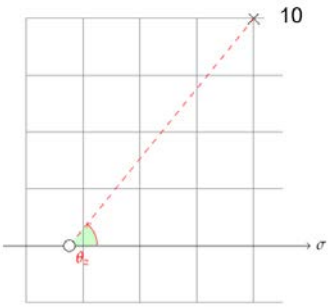


Imagen 58. Ubicación del cero introducido por el controlador PD

Conocida la deficiencia angular se calcula la posición del cero del controlador proporcional derivativo.

$$arctan\frac{10}{z-10} = DA = 50.71° \rightarrow z = 18.182 \quad (9.27)$$

Se determina la ganancia del controlador.

$$|CP(s)|_{s=Pd} = \frac{1}{k} \quad (9.28)$$

$$\left|\frac{9}{s(s+9)} \cdot (s + 18.182)\right|_{s=Pd} = \frac{9 \cdot \sqrt{(-10+18.182)^2+10^2}}{\sqrt{(-10)^2+10^2} \cdot \sqrt{(-10+9)^2+10^2}} = \frac{1}{k} \quad (9.29)$$

$$k = 1.222 \tag{9.30}$$

El controlador proporcional derivativo queda de la siguiente forma.

$$PD(s) = 1.22 \cdot (s + 18.182) \tag{9.31}$$

Por último, se comprueba que el diseño es correcto mediante la ecuación característica en lazo cerrado:

$$1 + PC(s) = s \cdot (s + 9) + 9 \cdot 1.22 \cdot (s + 18.182) = s^2 + 19.98s + 199.63836 \tag{9.32}$$

$$s = -10 \pm j10 \tag{9.33}$$

Los polos obtenidos en lazo cerrado coinciden con los polos deseados, dando por válido el diseño.

Una vez calculado el controlador proporcional derivativo se procede a calcular el controlador proporcional integral para eliminar el error en régimen permanente. Para realizar la sintonía del controlador basta con situar la pareja polo-cero, de forma que distorsione lo menos posible la dinámica del sistema.

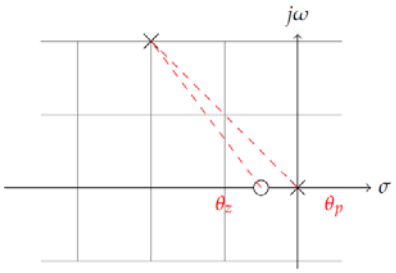


Imagen 59. Ubicación del polo-cero introducidos por el controlador PI

Se obtiene la posición del cero.

$$\theta_z - \theta_p = -5° \tag{9.34}$$

$$\theta_z = 180 - arctan\left(\frac{10}{10-z}\right) \tag{9.35}$$

$$180 - arctan\left(\frac{10}{10-z}\right) - 135° = -5° \rightarrow z = 1.609 \tag{9.36}$$

Se determina la ganancia del controlador.

$$|CP(s)|_{s=Pd} = \left|\frac{9}{s(s+9)} \cdot 1.22 \cdot (s + 18.182) \cdot \frac{s+1.609}{s}\right|_{s=Pd} = \frac{1}{k} \tag{9.37}$$

$$\frac{9 \cdot 1.22 \cdot \sqrt{(-10+18.182)^2+10^2} \cdot \sqrt{(-10+1.609)^2+10^2}}{\sqrt{(-10)^2+10^2} \cdot \sqrt{(-10)^2+10^2} + \sqrt{(-10+9)^2}} = \frac{1}{k} \tag{9.38}$$

$$k = 1.085 \tag{9.39}$$

De manera que el controlador PID(s) es:

$$PID(s) = 1.085 \cdot \frac{s+1.609}{s} \cdot 1.22 \cdot (s + 18.182) \tag{9.40}$$

$$PID(s) = 1.3237 \cdot \frac{(s+1.609) \cdot (s+18.182)}{s} \tag{9.41}$$

Para comprobar que el sistema es correcto y que los polos deseados están donde se desean se puede simular en *Matlab.*

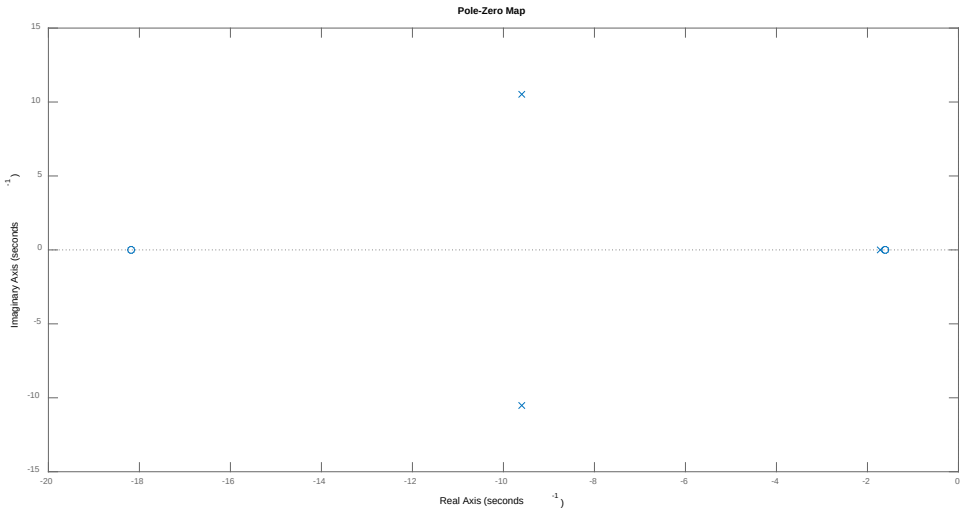


Imagen 60. Mapa de polos y ceros tras introducir el PID

Los polos deseados son *-10±j10*, como se puede observar, están desplazados ligeramente hacia la derecha debido a las aproximaciones en los cálculos. Pero el sistema responde según lo establecido. Su comportamiento se puede apreciar en la siguiente gráfica, que muestra la respuesta al escalón en lazo cerrado del controlador PID implementado junto con la planta de posición.

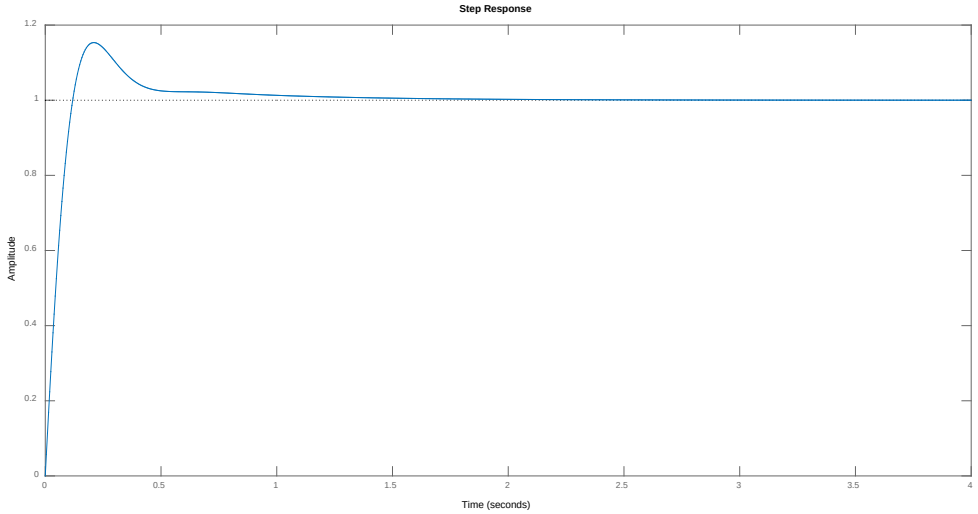


Imagen 61. Respuesta al escalón del lazo de control

9.3.2. Pruebas y controlador definitivo

Tras realizar un controlador teórico para la planta en la vida real se procede a implementar un controlador que permita mover al conjunto de una manera suave y precisa, llegando a donde se indica, como se ha comentado en la introducción de este punto, debido a que la planta de posición tiene un integrador, el error de posición va a

eliminarse, posibilitando el control con el empleo de una ganancia. En este apartado se prueban una serie de controladores hasta llegar al más adecuado.

La primera prueba es realizada con un controlador de ganancia 0.5.

$$C_p(s) = 0.5 \tag{9.42}$$

Observando el comportamiento del sistema con este controlador se apreció que no presentaba mucho error en la salida del sistema, pero era un poco lento, por lo que se decidió aumentar la ganancia.

El segundo controlador usado fue de una ganancia de 5, es decir, de 10 veces más que el primero, de esta manera la respuesta al sistema iba a ser más rápida.

$$C_p(s) = 5 \tag{9.43}$$

Se comprobó en el conjunto de la servoválvula, donde se tardaba mucho menos en reaccionar a variaciones en la referencia, pero al final no se estabilizaba. Cuando se acercaba al valor final, el engranaje seguía buscando la posición, no alcanzaba el destino deseado porque la ganancia era muy grande y el error final aumentaba. Por lo que se optó por reducir el controlador de nuevo.

La tercera prueba se basó en un controlador del tipo:

$$C_p(s) = 2 \tag{9.44}$$

Con una ganancia menor, la respuesta seguía siendo rápida y adecuada, pero al final seguía apareciendo el error que no permitía al sistema estabilizarse por completo. Visto esto y, observando que el error disminuía al disminuir la ganancia, se volvió a hacer otra prueba, esta vez con un controlador de ganancia unidad.

$$C_p(s) = 1 \tag{9.45}$$

En este caso la respuesta del sistema seguía siendo óptima y el conjunto iba sin problemas al valor final deseado, es decir, el engranaje dejaba de buscar la posición porque ya había llegado a ella y permanecía en ella hasta una nueva orden. Por lo que el controlador definitivo de posición se trata de un controlador de ganancia de tipo unidad.

9.3.3. Implementación de la ley de control de posición en Simulink

Como se explicó en la sección superior, se realizaron diferentes pruebas de controladores para el ajuste de posición, en este apartado se va a proceder a visualizar varias de ellas para justificar por qué se eligió el de ganancia unidad.

En primer lugar, se realizó un controlador de poca ganancia, pero se observó que respondía de manera más lenta que los demás, sus gráficas no están realizadas. El primer controlador comprobado en esta sección es el de ganancia 5.

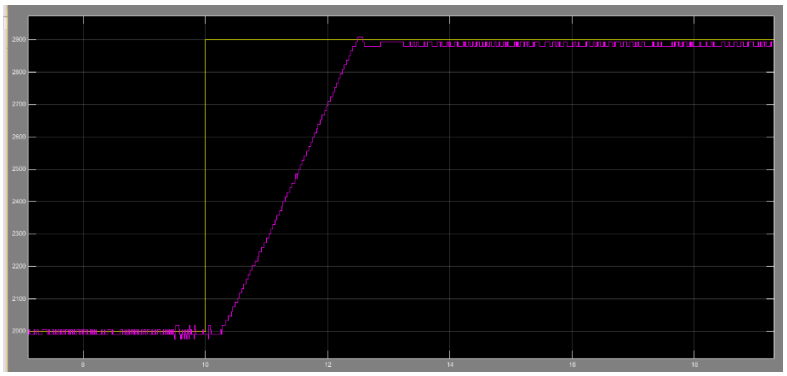


Imagen 62. Respuesta a la variación positiva de entrada (C(s)=5)

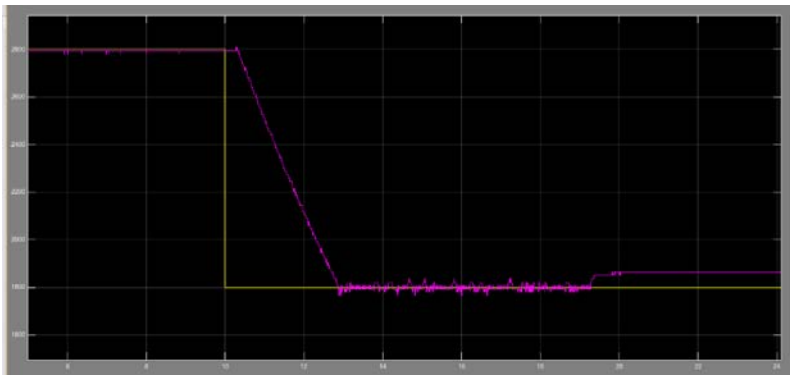


Imagen 63. Respuesta a la variación negativa de entrada (C(s)=5)

Como se puede apreciar en la segunda respuesta, este controlador no presenta una respuesta muy óptima, el sistema no se detiene al llegar al valor final, sino que permanece todo el tiempo un poco desviado e intentando llegar a él.

Debido a los errores presentados con este controlador se optó por probar otro de menor ganancia y observar cual presentaba una respuesta razonable y llegaba a la posición deseada. En el siguiente caso se visualiza el controlador de ganancia 2.

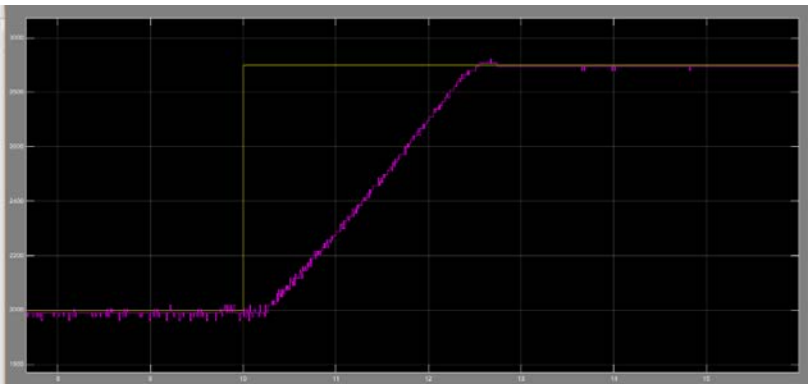


Imagen 64. Respuesta a la variación positiva de entrada (C(s)=2)

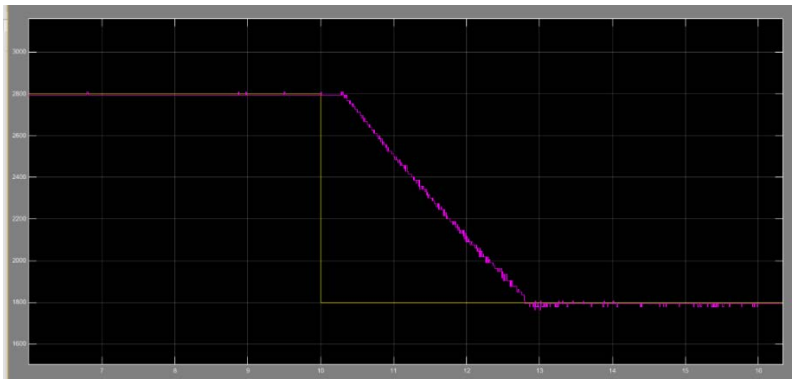


Imagen 65. Respuesta a la variación negativa de entrada (C(s)=2)

Aunque la respuesta parece óptima, el sistema seguía presentando oscilaciones a la salida, de manera que se optó por implementar un controlador de ganancia unidad porque su comportamiento era el más adecuado de todos.

Al usar el controlador de ganancia unidad, el sistema llega al valor indicado sin mostrar ningún comportamiento extraño en el recorrido, además se trata de un movimiento más suave, con menos cambios de velocidad y, como se puede observar en las representaciones de a continuación, es un proceso muy preciso.

Las respuestas del sistema utilizando un controlador de ganancia unidad quedan de la siguiente manera.

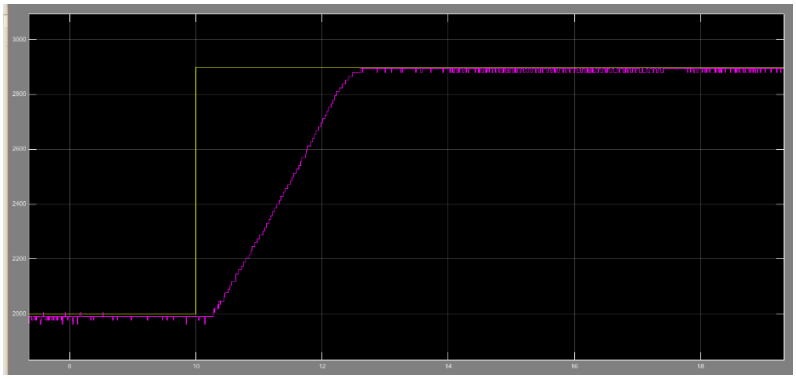


Imagen 66. Respuesta a la variación positiva de entrada (C(s)=1)

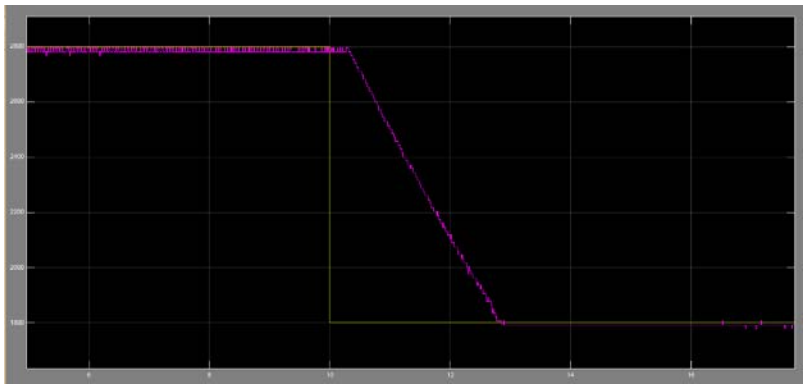


Imagen 67. Respuesta a la variación negativa de entrada (C(s)=1)

Como se puede observar en las respuestas, el sistema llega de manera muy precisa a la posición deseada en un período razonable de tiempo. Además, el conjunto de engranajes reacciona muy bien a las variaciones de entrada introducidas, deteniéndose cuando de manera suave cuando se llega al valor deseado.

10. Creación de la PCB

10.1. Introducción

Un circuito impreso (*PCB* en inglés), es una placa o tarjeta utilizada para realizar el emplazamiento de distintos elementos que forman parte del circuito y de las conexiones eléctricas existentes entre ellos.

Antes era usual la fabricación de circuitos impresos para el diseño de sistemas mediante técnicas caseras, sin embargo, esta práctica ha ido disminuyendo con el paso del tiempo. En los últimos años ha ido reduciéndose el tamaño de los componentes electrónicos de manera considerable, lo que implica que la separación entre pines sea menor para circuitos integrados de alta densidad. Debido a las actuales frecuencias de operación de los dispositivos, es requerida y necesaria una muy buena precisión en el proceso de impresión de la placa con la finalidad de garantizar tolerancias mínimas.

Los circuitos impresos más sencillos son los que contienen caminos de cobre (*tracks*) solo por una de las superficies de la placa. A dichas placas se les conoce como circuitos impresos de una capa (cuya traducción al inglés es "*1 Layer PCB*").

Sin embargo, los circuitos impresos más comunes hoy en día son los que contienen 2 capas (*2 Layer PCB*). Aunque, dependiendo de la complejidad del diseño físico del circuito diseñado (*PCB layout*), pueden llegar a fabricarse hasta de 8 o más capas.

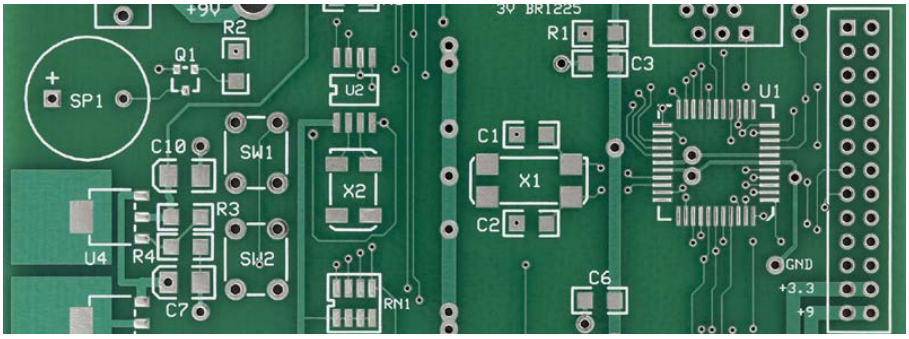


Imagen 68. Ejemplo de PCB

10.2. Diseño de la placa en Kicad 4.0.7

El programa utilizado para crear el circuito impreso propio para el sistema de control de servoválvula es *KiCad. KiCad* es una herramienta software con la característica de ser *open-source* que se utiliza para la creación de diagramas electrónicos y para el diseño de placas de circuitos impresos. *KiCad*, bajo su estructura, incorpora un conjunto que

presenta las herramientas de software independientes necesarias para el diseño de la *PCB*.

Nombre del programa	Descripción	Extensión de Archivo
KiCad	Gestor de proyectos	*.pro
Eeschema	Editor de esquemas (esquemas y componentes)	*.sch, *.lib, *.net
CvPcb	Selector de huellas	*.net
Pcbnew	Editor de placa de circuito	*.kicad_pcb
GerbView	Visor de ficheros Gerber	Extensiones gerbers usuales
Bitmap2Component	Crea componentes o huellas a partir de imágenes bitmap.	*.lib, *.kicad_mod, *.kicad_wks
PCB Calculator	Calculadora de componentes, tamaños de pistas, espaciado eléctrico, códigos de colores, y más…	Ninguno
PI Editor	Editor de formatos de página	*.kicad_wks

Tabla 4. Herramientas de software independientes de KiCad

KiCad no presenta limitaciones en cuanto al tamaño de la placa del circuito, puede gestionar de forma sencilla hasta 32 capas de cobre, 14 capas técnicas y hasta 4 capas auxiliares. Con *KiCad* es posible crear todos los archivos necesarios para la construcción de placa.

Para comenzar con el diseño de la placa se deben elegir los elementos necesarios que introducir en ella. La primera característica que se ha de conocer es que el sistema de la servoválvula está conectado a la maqueta mediante un conector DIN de 7 pines, tal y como se explica en la parte de Anexos, a la hora de hablar sobre las maquetas.

Las salidas que presenta este conector son las siguientes:

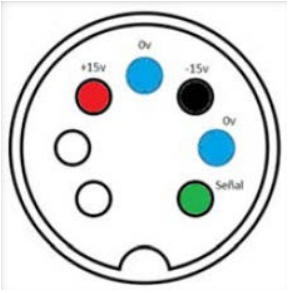


Imagen 69. Conexiones del conector DIN de 7 pines

10.3. Primeros diseños de la placa

Para el funcionamiento de la servoválvula se utiliza la fuente de alimentación positiva, es decir, los pines que aportan 15 voltios y tierra. La señal de regulación de la servoválvula llega del

pin llamado "*Señal",* el cual envía un dato procedente de los valores de nivel que presentan los depósitos en la maqueta, indicando si se tiene que abrir más o menos.

La entrada de alimentación, como se ha explicado, es de 15 voltios, mientras que el motor de la servoválvula trabaja con 12, por lo que es necesario introducir una etapa reguladora de tensión que sea capaz de transformar esos 15 voltios de entrada en 12 voltios con el fin de que el motor funcione según lo previsto.

Para la etapa reguladora se escoge un regulador capaz de realizar esta función, este tipo de dispositivos están normalizados y se encuentran en el mercado, se trata del *LM7812*.

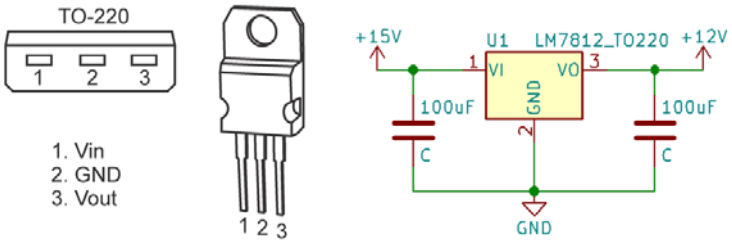


Imagen 70. Regulador LM78xx

Para la utilización de este integrado se necesitan muy pocos componentes adicionales, usando condensadores para filtrar y estabilizar la tensión de entrada y de salida. Este regulador *LM7812*, en concreto, transforma la señal de entrada en un valor de 12 voltios a la salida y la corriente de salida que suele ofrecer es de 1A. Hay que tener en cuenta que el voltaje de entrada debe de ser, como mínimo, 2.5V ó 3V superior a la salida para conseguir que el integrado funcione perfectamente. Al regulador le llegan las señales de un bornero de dos entradas introducido para conectar los cables que llegan de la fuente de alimentación y que son más gruesos que los normales, por lo que no valen conectores hembra normales.

El *driver L298N* empleado para el funcionamiento del sistema de mantiene, como se ha comprobado que funciona correctamente se ha tomado esta decisión para agilizar el proceso y para asegurar el buen funcionamiento del conjunto. Los cables entre el *driver* y la *PCB* se adhieren a la placa por medio de conectores hembra. Mientras que los que van desde el motor al driver no se introducen a la *PCB*, van independientes de la placa. A la tira de 7 pines implementada para conectar la *PCB* con el *driver* le llegan los doce voltios a uno de los pines, la tierra del sistema a sus dos siguientes, una señal de 5 voltios al cuarto, para alimentar el regulador del *driver*, y los tres últimos pines son los que controlan tanto la velocidad como el sentido del motor (*IN1*, *IN2* y *ENA*).

Para el potenciómetro multivuelta solidario al movimiento del motor se vuelven a conectar sus terminales a 3 conectores hembra, el primer y el tercer pin son la alimentación del potenciómetro, mientras que el segundo se conecta a una entrada analógica para poder medir el valor en cada caso.

Para la señal de entrada se crea un sistema basado en un jumper, con el jumper se podrá decidir si se desea que la referencia llegue procedente de la señal del sistema o del potenciómetro usado hasta ahora para simular la entrada.

De tal forma que el programa principal queda de la siguiente manera.

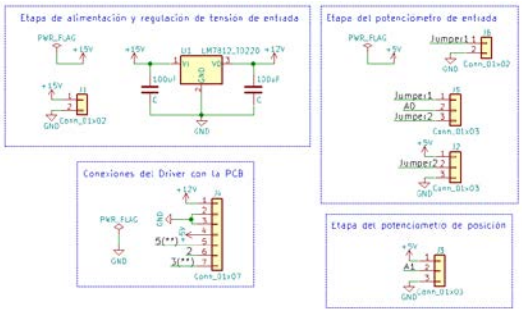


Imagen 71. Esquema de los componentes introducidos en Kicad

Básicamente, este es el eje principal de las diferentes placas que se han diseñado en *KiCad*, varias son las pruebas que se realizaron para comprobar que los componentes estaban situados correctamente en la placa y que sus pines cabían en los agujeros generados para ello. Las placas microcontroladores utilizadas en el diseño son *Arduino Uno* y *Arduino Nano*, para los cuales se introdujeron las huellas y las librerías correspondientes, diferentes fueron las mediciones realizadas durante el proceso de acondicionamiento de la servoválvula, también fueron diferentes las ideas para el diseño de la placa, así como su disposición, a continuación, se muestran una serie de pruebas antes de elegir la definitiva.

Como se ha comentado, tras varias pruebas se optó por aquellas en las que se incorporó el jumper que permitía seleccionar el potenciómetro o la señal del sistema. En primer lugar, se pensó en introducir el potenciómetro en la placa, pero luego se concluyó que era mejor introducir una serie de pines para, mediante cables, conectar de manera externa el potenciómetro para que no ocupara espacio en la placa.

10.3.1. PCB con Arduino Nano

La mejor idea para implementar en el sistema de la servoválvula es introducir una PCB con el *Arduino Nano*, este microcontrolador tiene la característica de tener un tamaño muy reducido y, además, su programación y sus componentes son iguales que los de *Arduino Uno,* por lo que no es necesario modificar nada de los archivos de *Matlab & Simulink* empleados para dicha programación.

La placa diseñada, con su estructura, responde muy bien a los componentes del sistema utilizados en un espacio pequeño, muy óptimo para implementarlo. Para la placa era necesaria la huella de *Arduino Nano* y se introdujo tanto la huella como la librería en el software.

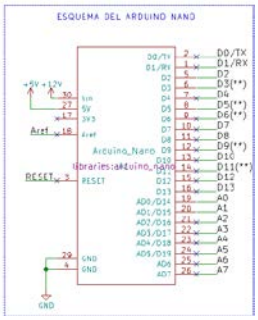


Imagen 72. Arduino Nano en el esquemático de KiCad

Con el esquemático ya creado se procede a crear la placa dibujando pistas en la herramienta de *KiCad* llamada *PCBnew.* El esquemático está formado por el *Arduino Nano* y por los diferentes elementos que había que añadir a la placa para que funcione como se desea. Una vez dibujadas todas las pistas se observa que se pueden realizar todas las uniones sólo en una capa, simplificando en gran medida el proceso debido a que se evita el salto entre capas. La placa diseñada para implementar el *Arduino Nano* tiene la siguiente forma.

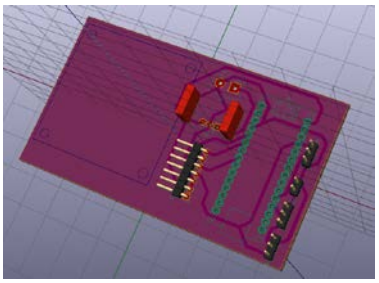


Imagen 73. Visualización en 3D de la PCB para Arduino Nano

En la imagen, el cuadrado azul con los círculos dibujados en cada uno de sus extremos representa el driver, el cual va implementado en esa parte de la placa y la idea es atornillarlo a ella para fijarlo.

10.3.2. PCB con Arduino Uno

Con la placa de *Arduino Nano* diseñada para una posterior impresión, se opta por realizar otra muy similar, pero, en este caso, para el *Arduino Uno*, toda la programación y todas las pruebas ejecutadas para el control del sistema se han realizado con este microcontrolador, teniendo constancia y seguridad de que todo funciona correctamente.

Sabiendo esto y midiendo el espacio que se tiene en el sistema para introducir todos los componentes necesarios se comprueba que, en unas determinadas posiciones, todo el conjunto formado por el *Arduino Uno,* el *driver L298N* y el resto de los componentes tiene espacio suficiente.

El diseño de la placa adquiere un mayor grado de dificultad porque se debe tener muy claro dónde colocar los elementos de forma que todo quede de la mejor manera posible. Al observar el espacio y deduciendo que el *driver* no tenía por qué ir implementado en la placa se decidió colocar el *driver* y el sistema compuesto por el *Arduino* y la *PCB*. Así, mediante cables se unen ambas partes y el problema del espacio se reduce.

La estructura de la *PCB* es semejante a la creada con el *Arduino Nano*, lo que cambia son las dimensiones y la huella, que en este caso hay que utilizar tanto la huella como la librería para trabajar con la placa empleada.

En este caso, la huella de *Arduino Uno* delimita las dimensiones de la *PCB* a realizar, para crearla se tienen muy en cuenta sus medidas ya que se desea que ocupe unas dimensiones parecidas a las de la propia placa o menores.

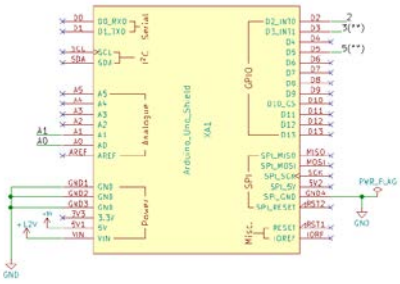


Imagen 74. Arduino Uno en el esquemático de KiCad

En primer lugar, se piensa en diseñar una placa para *Arduino Uno* que contenga el *driver,* el cual va insertado con unos tornillos a la base de la *PCB,* y que también incluya el potenciómetro que sirve de referencia para controlar el movimiento del sistema. De esta forma se van creando las primeras ideas y prototipos.

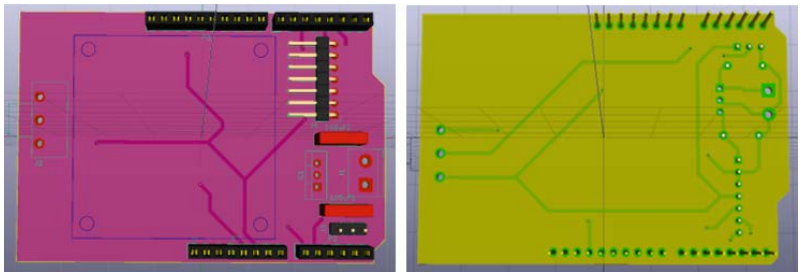


Imagen 75. Primer diseño para Arduino UNO

Los tres agujeros de la izquierda hacen referencia al potenciómetro que puede implementarse en esa parte a la placa, de tal forma que se controlaría la posición desde ese punto, de la misma manera que en esos huecos puede ir soldado el potenciómetro, también sirven para introducir en ellos la señal del sistema que nos mande las entradas correspondientes.

El siguiente diseño consiste en centrarse en recibir los valores del sistema solamente desde su señal, sin incluir ningún potenciómetro que haga la función de entrada para probar y observar que la placa funciona correctamente. En este caso, se omiten los espacios creados para introducir el potenciómetro y se introducen unos machos para soldar la señal procedente del sistema. Como se puede ver en la siguiente imagen y en el anterior diseño, existe un recuadro azul y unos círculos que hacen referencia a dónde debe ir colocado el *driver* para no entorpecer ninguna pista. El contorno azul no se va a imprimir, sólo es para ordenar los elementos y las pistas de manera que cuando se introduzca el *driver* no haya ningún problema.

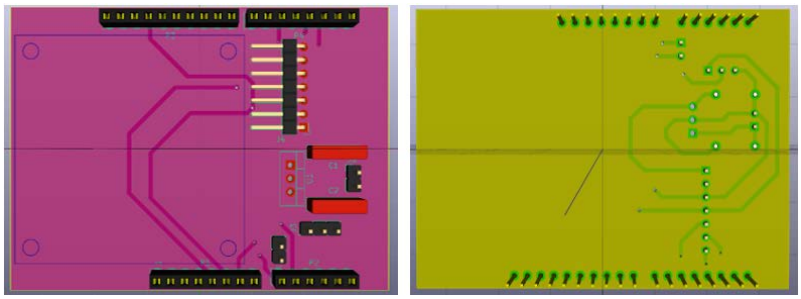


Imagen 76. Segundo diseño para Arduino Uno

El tercer diseño de *PCB* creado vuelve a tener pistas por las dos caras, su configuración no permite realizar todo el conexionado en una única capa debido a la conexión de machos, que se sueldan en la capa superior de la placa, y las hembras, con las que hay que hacer lo propio, pero por debajo, tal y como sucede en los anteriores diseños para *Arduino Uno*. De forma que la placa creada en tres dimensiones presenta la siguiente estructura.

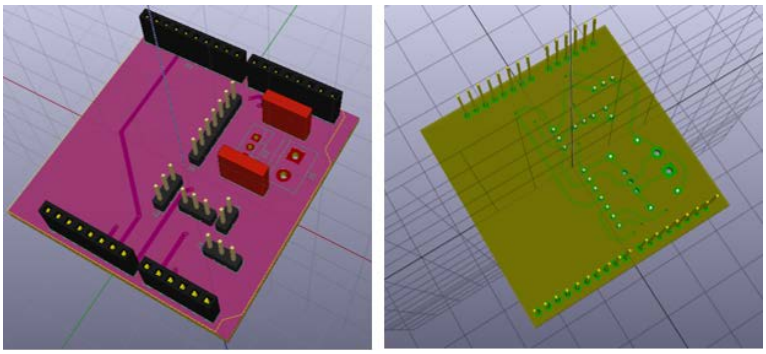


Imagen 77. Visualización en 3D de la PCB para Arduino Uno

Una vez construida la placa se mide la continuidad entre todas las soldaduras realizadas y las pistas de cobre, asegurando así la circulación de corriente entre todos los componentes y las pistas creadas. Luego se prueba en la placa *Arduino Uno* y se observa el perfecto funcionamiento, de tal manera que todo funciona según lo previsto.

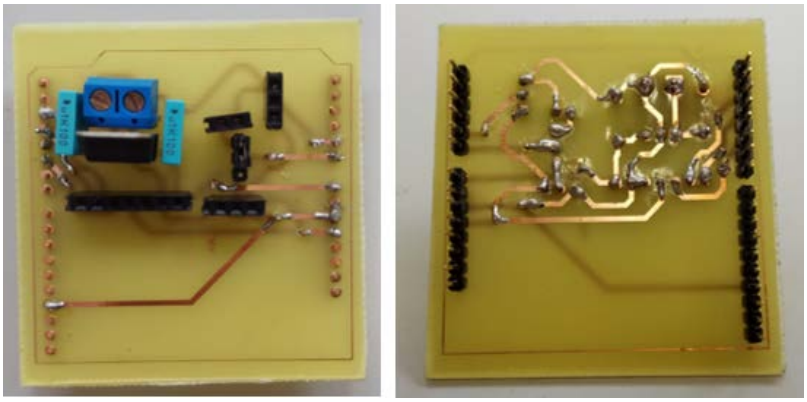


Imagen 78. Primera placa montada

A la hora de realizar un primer montaje del prototipo en el sistema de servoválvula, se observó que los cables se reducían, ya que todos los que conectan con *Arduino* se eliminan. El problema reside en los cables que conectan el *driver* con la placa para permitir el funcionamiento del sistema, como se puede apreciar, estos cables son varios y si se quisiese implementar la *PCB* creada en el circuito habría que acondicionarlo de manera que todo quedase de una manera discreta y perfectamente conectado. Esta placa es una opción, viable, si se acondiciona y las conexiones son seguras.

En la siguiente foto se ve la primera implementación del prototipo, esta implementación sirvió para comprobar el buen funcionamiento que tenía del sistema y hacerse una idea de cómo podría quedar y ser el resultado final, aunque las opciones barajadas para la posición de los dispositivos son varias.

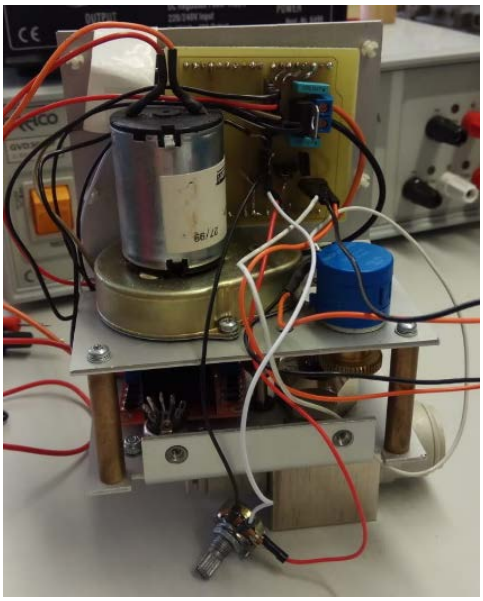


Imagen 79. Implementación de la PCB en el sistema

Una vez tenida esta primera opción, se procede a crear una placa más sofisticada que nos permita reducir los cables que se presentan en este primer prototipo.

10.4. Segundo diseño de la placa

La primera opción tiene el inconveniente de la existencia de varios cables que conecten el driver con el resto del circuito, aunque su funcionamiento es bueno, responde muy bien a las variaciones en la entrada. En esta segunda opción se opta por una estructura más compleja, sin utilizar el *driver L298N* comprado, sino insertándolo en la placa a realizar para eliminar todos esos cables que conectan el *driver* comercial con la *PCB* construida.

Para llevar a cabo el diseño se utiliza en este caso, además de la huella para el tipo de *Arduino* que queramos implementar, los componentes, librerías y huellas propias del doble puente H (*L298N*). Estas huellas se pueden encontrar de manera fiable buscando por internet y conociendo la existencia de una plataforma muy fiable para descargar este tipo de archivos (*GitHub).*

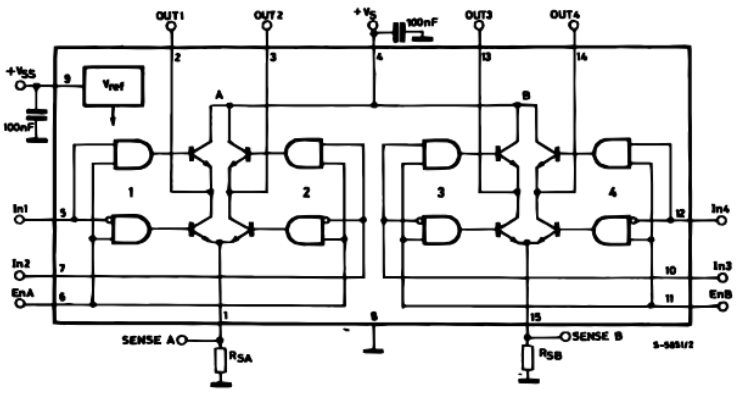


Imagen 80. Doble puente H del driver L298N

Para la construcción de esta segunda placa, los componentes a utilizar son los mismos que para la anterior ya que seguimos necesitando una etapa de entrada a la placa, así como una sección que se encargue de la regulación y pines para conectar los diferentes elementos. A estos componentes se les va a incorporar el doble puente H ya comentado, este dispositivo tiene la característica de poder controlar hasta dos motores de corriente continua (opciones A y B), pero en el sistema sólo se desea gobernar el comportamiento de un único motor DC. Por lo que sólo se elegirá uno de los dos puentes H que ofrece el chip, utilizando los pines correspondientes.

Teniendo en cuenta la disposición de los pines del *L298N* se puede realizar el circuito equivalente que permite controlar de forma bidireccional el motor de la servoválvula.

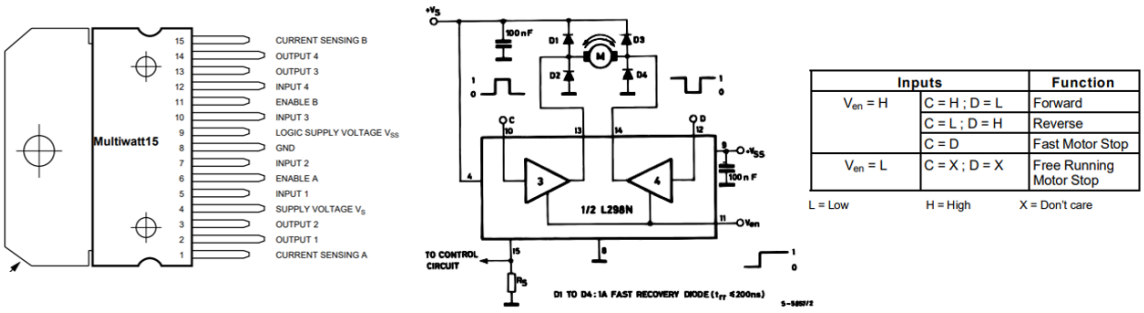


Imagen 81. Configuración B del Driver L298N

En el proyecto se escoge la opción A, que tiene la misma configuración que la B pero usando el otro puente H que posee el dispositivo, así como sus respectivos pines. De tal manera que la parte incorporada al esquemático de *KiCad* con *Arduino Uno* es la siguiente.

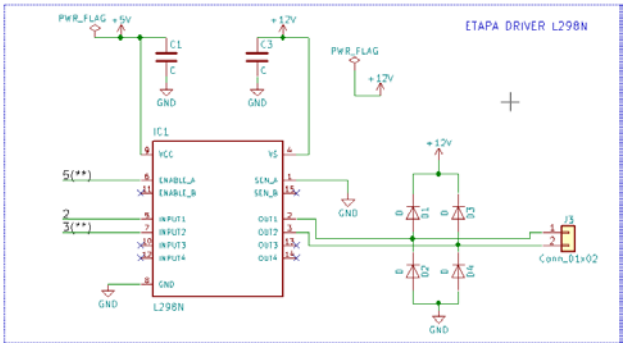


Imagen 82. L298N en KiCad

Como se puede observar, se escoge la opción A del doble puente H para gobernar el motor deseado, se utiliza un puente de diodos para poder controlar el sentido del motor en cada caso y con el *enable* se controla la velocidad en cada momento. El *driver* comercial llevaba implementado además un regulador, el cual quedaba deshabilitado cuando se desean controlar motores mayores a 12V, como en las diferentes pruebas realizadas no se ha contado con este dispositivo, se ha optado por no implementarlo en el diseño final del circuito integrado.

Una vez creado el esquemático se representa en 3D como se desea que quede la placa, con sus componentes ya distribuidos y las pistas dibujadas.

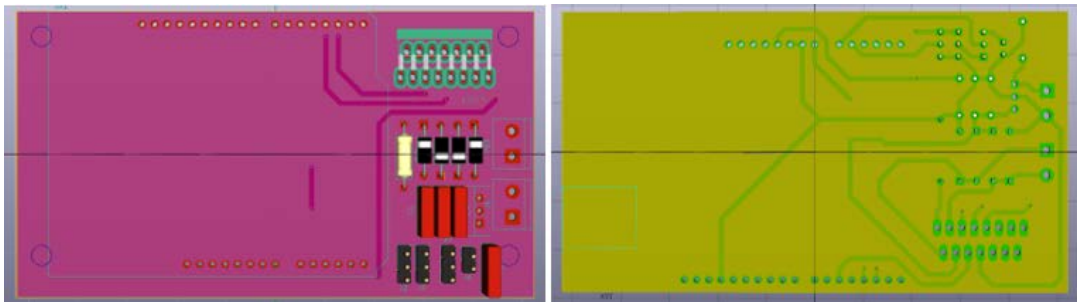


Imagen 83. Placa con el Driver L298N Integrado

Tras el diseño de esta placa se procede a construirla de igual forma que la anterior, como se puede observar, los componentes necesarios para la creación de la placa son cuatro diodos, una resistencia y cuatro condensadores, tal y como aparece indicado en el esquemático que permite dibujar las pistas y comprobar si todas las conexiones se han realizado de manera correcta y que no se deja ningún dispositivo sin conectar, en esta característica la herramienta *KiCad* es muy útil, ya que te va indicando cada vez que elementos quedan por ser conectados todavía.

Si se recuerda, el esquemático completo queda de la siguiente manera, sufre una modificación respecto a la anterior placa, tal y como se ha comentado, ya que se inserta el *driver L298N* en el sistema, pero el resto de las vías y conexiones se mantienen.

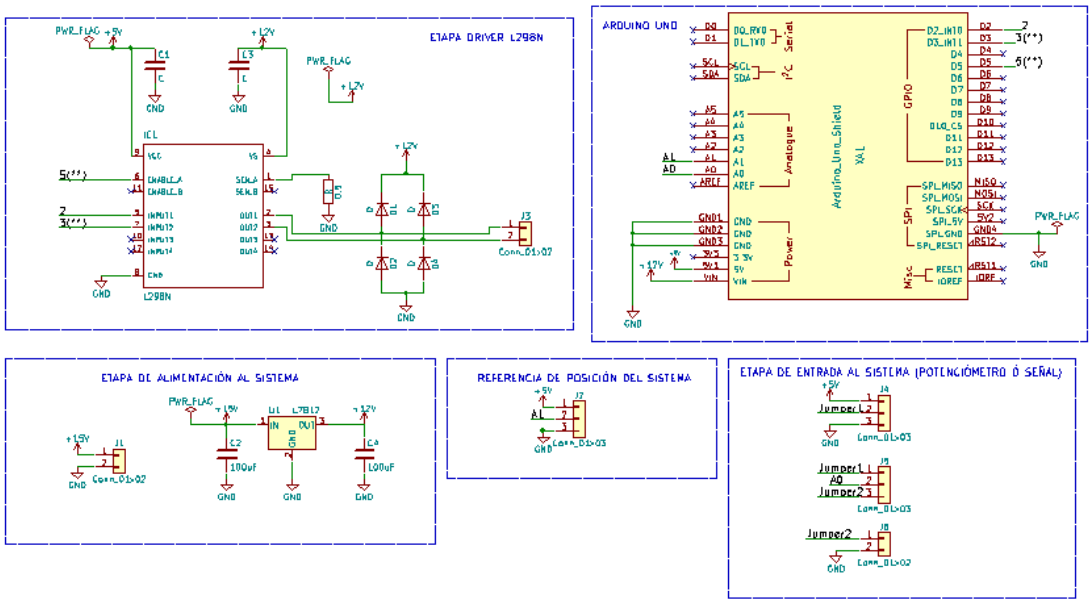


Imagen 84. Esquemático creado para la realización de la placa

Tras construir la *PCB* la disposición de los elementos queda de la siguiente manera, dejando al lado izquierdo todo lo perteneciente a implementar el *Arduino Uno* y en el derecho se introducen todos los demás dispositivos necesarios.

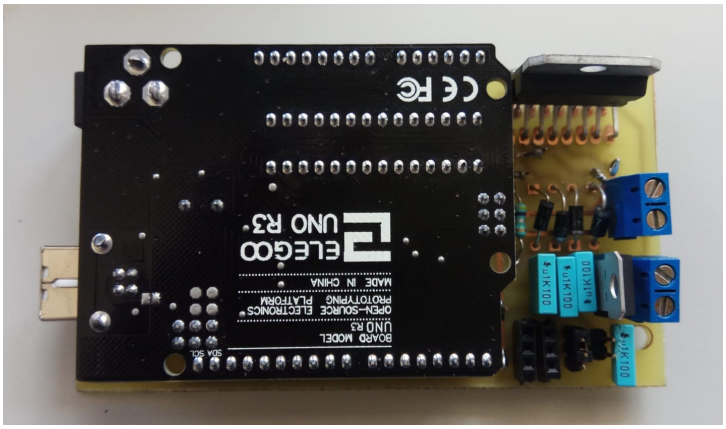


Imagen 85. Etapa de construcción y soldadura de la PCB con el driver implementado

El esquemático y el diseño de esta *PCB* son correctos*, KiCad* no presenta ningún error ni nada no conectado, además la construcción del circuito integrado va según lo previsto, realizando los pasos con cuidado y de manera adecuada, desde imprimir los planos en acetato y llevarlos a la placa hasta todo el proceso de insolación llevado a cabo. Bañando primero en cobre la placa con el circuito impreso ya implementado sobre ella y luego sumergiéndola en una disolución de uno es a uno de agua, agua oxigenada y agua fuerte, consiguiendo así que en todos los lugares de la placa donde no existan pistas de cobre desaparezca dicho metal.

11. Conclusiones

Este trabajo ha servido para demostrar como las nuevas tecnologías facilitan el control de diferentes dispositivos que, a priori, parecen laboriosos y complejos. En este caso se ha gobernado un sistema de servoválvula pero puede servir para cualquier dispositivo del que se quiera controlar su comportamiento.

Las opciones para realizar el proyecto han sido múltiples, si se hubiese dado el caso, una alternativa muy a tener en cuenta hubiese sido utilizar el *Arduino Nano*, que es semejante al *Arduino Uno,* pero con menos pines y de reducido tamaño. Pero se ha trabajado con el *Arduino Uno*, una placa muy sofisticada y que engloba todo lo necesario para realizar proyectos de esta escala e incluso de mayores magnitudes.

Actualmente la tecnología y la ingeniería están viviendo una época muy buena, con el desarrollo de impresoras 3D, de diferentes robots, placas microcontroladoras, es muy fácil y barato crear y diseñar lo que se quiera. Como ha sucedido en este proyecto, en el cual se han impreso en 3 dimensiones instrumentos necesarios como la corona circular o el soporte del encoder.

En el control de velocidad se han presentado varios problemas, se observó como el sistema no leía bien los pulsos y no se ejecutaba en tiempo real, por lo que el experimento no era válido. Una forma de solucionarlo fue modificar el número de huecos de la corona circular para asegurar un mayor ángulo entre pulso y pulso y que la lectura fuese clara. Mientras que para que el sistema se ejecutara y leyera los valores en tiempo real se probó con diferentes períodos de muestreo hasta conseguir lo deseado con uno de ellos (Ts=0.01s).

Continuando con el control de velocidad, como su planta es una ganancia pura, no es necesario añadirle un cero al sistema. Si se le añade el cero, de un PI (controlador Proporcional Integral) por ejemplo, va a dar más problemas que implementar un integrador y su ganancia. Esto es debido a que los ceros se traducen en sobreimpulsos y, si son dominantes, es algo que no se desea.

En el control de posición el problema que se encontró fue el del potenciómetro que indicaba la posición del sistema, como se ha explicado en la memoria, hubo que acondicionarlo debido a que una de sus patillas no funcionaba, la solución fue sencilla, ya que bastó con añadir una resistencia en serie a una de las dos que funcionaba para crear un divisor de tensión. Una vez solucionado esto y conociendo que la relación de engranajes entre las ruedas dentadas del sistema era de 1 (ya que las tres ruedas tenían 48 piñones cada una), se pudo hacer el control de posición de forma adecuada.

Con la construcción del primer circuito integrado se observó una reducción de cables considerable, ya que se dejaba de utilizar la *protoboard* y todo lo que conectaba con *Arduino Uno* se llevaba mediante pistas de cobre, pero los cables que conectaban el *driver L298N* con la placa microcontroladora seguían existiendo, por lo que, siendo el primer diseño creado una opción para implementar en el sistema. Se decidió crear otra *PCB* más sofisticada, que incluyese el *driver* para así eliminar las conexiones comentadas.

Gracias a la creación de este segundo circuito integrado se introdujeron todos los elementos necesarios en una placa, de manera que ocupasen el menor espacio posible,

de esta forma, y teniendo la tarjeta de *Arduino* ya programada, es necesario emplear muy pocos cables, lo que optimiza el proceso y permite que el resultado final sea fiable.

12. Vías de continuación

Una vez realizada la segunda placa descrita, la cual reduce en gran medida el número de cables que se tenía con la primera *PCB* realizada, construida e implementada en el sistema, faltaría conectarla en la planta real.

Tras conectar esta *PCB* en la primera servoválvula que tiene la planta de control de procesos *Feedback Procon 38-003* habría que hacer otra semejante para la segunda servoválvula de la maqueta. Como ambos sistemas son iguales y sus componentes (motor *Crouzet 82861010,* potenciómetro *Bourns* de 10kΩ y válvula de aguja) también, el control realizado con *Arduino* es exactamente el mismo, por lo que no habría que tocar nada de la programación ni del diseño de la placa, sólo habría que construir otra semejante para hacer funcionar al segundo dispositivo de igual manera que al primero.

Otra vía para continuar con el proceso es realizar el control del sistema de nivel/caudal. A la servoválvula le llega un valor de tensión que hace referencia al nivel de agua existente en el depósito de la maqueta, este nivel de tensión es el que determina el comportamiento del sistema, abriendo o cerrando la válvula de aguja según tenga que dejar pasar más agua para llenar el depósito o menos porque ya tiene suficiente.

Por último, después de toda la programación y el control realizados, sólo faltaría implementar la servoválvula en dicha maqueta, conectándola a los tubos de inserción rápida que el sistema tiene para introducir al dispositivo en su sitio. Cerrando así el lazo y el circuito de la planta de control de procesos y pudiendo poner en marcha todo el conjunto.

“DISEÑO, DESARROLLO Y CONSTRUCCIÓN DE SISTEMA DE CONTROL PARA SERVOVÁLVULA”

DOCUMENTO Nº 3 ANEXOS

Peticionario:	Universidad de La Rioja
Informantes:	Miguel de Gregorio Santamarina
	Estudiante de Ingeniería Electrónica Industrial y Automática

ÍNDICE

ANEXO 1. Planta de control de procesos FEEDBACK 38-003

El sistema *FEEDBACK 38-003* es una planta dedicada al estudio de técnicas de control de procesos industriales que requiere un PC corriendo bajo Windows. Entre los procesos que se pueden controlar están el de nivel, el de caudal y el de temperatura.

Requiere:

- PC corriendo bajo Windows, con un puerto de red libre o alternativamente un *switch* a través del cual sea posible conectarse en red al PC con el controlador de la planta del proceso.
- Plataforma de software *ESPIAL 93-420* sobre la que se carga los contenidos electrónicos provistos con este sistema y que se encarga de la comunicación entre equipos. Sólo se requiere una licencia ESPIAL por PC, sin que importe el número de paquetes y equipos específicos que se monten sobre ella.

Imagen 86. Planta 38-003

El sistema está constituido por:

- Planta para la enseñanza de Control de Procesos con control de nivel y caudal *38-100.*
- 2 interfases de proceso *38-200.*
- 2 controladores de procesos (digitales de 3 términos) *38-300.*
- Conjunto para la medición de caudal *38-420.*
- 2 módulos de display digital p/4-20 mA *38-490.*
- Planta de proceso p/temperatura (requiere *38-440*, *38-200* y *38-300*) *38-600.*
- Paquete de software Discovery para nivel/caudal y temperatura *38-901M*.

1. Principales elementos del módulo

1.1. Módulo Feedback de nivel y flujo PROCON 38-100

El módulo *FEEDBACK* de nivel y flujo *PROCON 38-100* es una planta para la enseñanza de Control de Procesos con control de nivel y caudal. Se trata de un sistema de un único lazo que permite el estudio de los principios del control de procesos, usando el nivel de líquido y tasas de flujo como las variables medidas en el proceso. Este sistema es completamente independiente, se trata de un circuito de agua a baja presión que fluye en una estructura acoplada a un panel, con un flujómetro ubicado en la parte inferior izquierda de la tubería.

Imagen 87. Módulo FEEDBACK 38-100

El módulo consta de un tanque superior de doble compartimiento enlazado a un tanque de desagüe por medio de válvulas manuales y solenoides. El agua es bombeada por el sistema a través del medidor de flujo y de la servoválvula. El nivel es medido en el tanque superior mientras que al flujo lo mide un flujómetro ubicado en la parte inferior izquierda de la tubería.

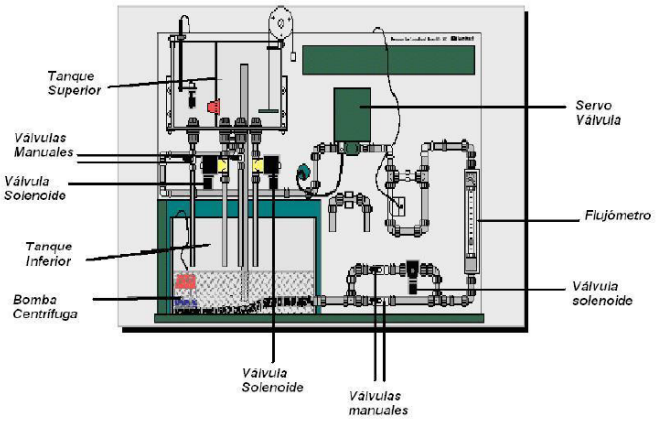


Imagen 88. Principales elementos de la maqueta FEEDBACK 38-100

Esta planta forma parte de varios conjuntos: *38-001*, *38-003*, *38-009* y *38-010*. Incluye:

- Tanque de reserva de agua.
- Tanque de proceso de 2+3 litros, dividido en 2 secciones por un tabique, con un agujero en la parte inferior que permite hacerlas solidarias.
- Descarga de rebalse para tanque de producto.
- Bomba centrífuga sumergida, 16 litros/minuto.
- Rotámetro (para la lectura visual del caudal) 0,4 a 4,4 litros/min.
- Servoválvula motorizada, con control 4-20 mA.
- 3 válvulas solenoide de 24 V.
- 4 llaves de paso manuales.

El circuito hidráulico es reconfigurable y permite a los usuarios implementar otros elementos como:

- Caudalímetro.
- Sensores de nivel *ON/OFF* proporcionales
- Etc.

1.2. Interfaz de proceso Feedback 38-200

Se trata de una unidad de servicios comunes para las plantas de la familia *Procon Feedback 38-xxx*.

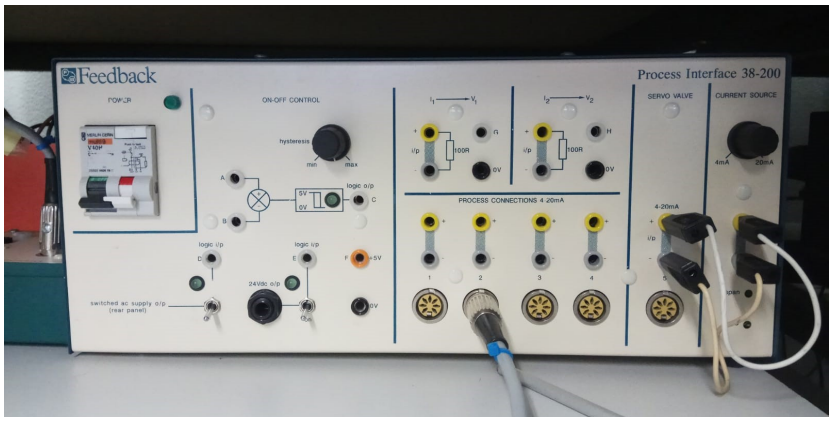


Imagen 89. Interfaz de proceso Feedback 38-200

Esta interfaz de proceso incluye:

- Alimentación de baja tensión para sensores, acondicionadores y actuadores.
- Comparador con histéresis variable, rango de entrada: 0-5 V.
- Buffer con entrada lógica, salida de potencia de 24 voltios y override manual.
- Buffer con entrada lógica, salida por relé para dispositivos externos y override manual.
- Alimentación y tratamiento de señales para 4 dispositivos de entrada 4-20 mA.
- Alimentación y salida de señal 4-20 mA para control de servoválvula compatible.
- 2 conversores de corriente (4 a 20 mA) a tensión a través de resistor de carga de 100 Ohm.

- Fuente de corriente regulable 4-20 mA para generar set-points o simular el ingreso de una señal de sensor.
- Pachera de conexión con fichas banana hembra de 4mm.

Esta interfaz de proceso está conectada al sistema y proporciona todas las alimentaciones necesarias para el módulo, sensores y controlador. Suministra una fuente de corriente de 4 a 20 miliamperios y dos fuentes de voltaje; positiva y negativa.

La interfaz está conectada el sistema de servoválvula por medio de un conector DIN de 7 pines. La configuración de las conexiones se muestra a continuación.

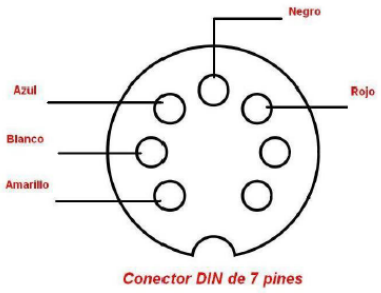


Imagen 90. Conector DIN de 7 pines

1.3. Conjunto para la medición de nivel Feedback 38-400

Se monta sobre la planta anteriormente descrita (*38-100*), para ser alimentada se conecta a la interfaz de procesos *38-200*. Incluye:

- Sensor de nivel *ON/OFF* con altura ajustable.
- Sensor de nivel proporcional, con flotador, polea y potenciómetro multivuelta industrial.
- Caja acondicionadora de señal con base magnética, salidas lógicas por el medidor *ON/OFF* y salida de 4 a 20 mA por el medidor proporcional.

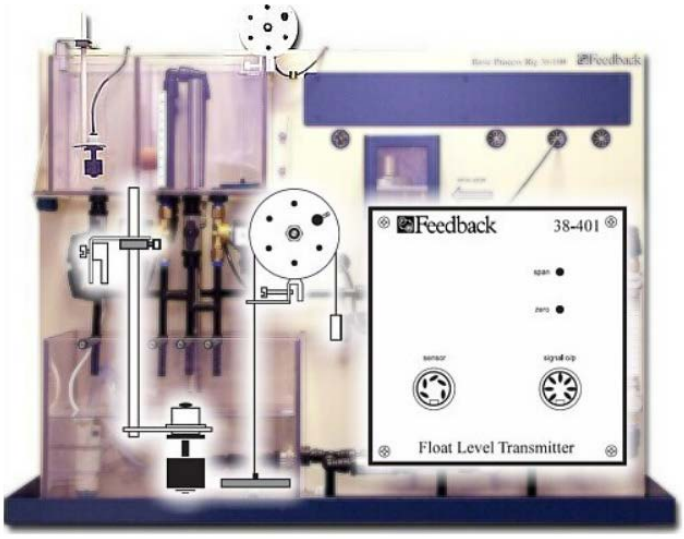


Imagen 91. Conjunto medición de nivel feedback 38-400

1.4. Conjunto para la medición de caudal Feedback 38-420

Este sistema se monta, al igual que el otro conjunto, sobre la planta *Feedback 38-100*. Toma su alimentación de la interfaz de procesos *38-200*. Incluye:

- Turbina medidora de caudal 0-4.5 litros/min.
- Caja accionadora de señal con base magnética y salida de 4 a 20 mA.

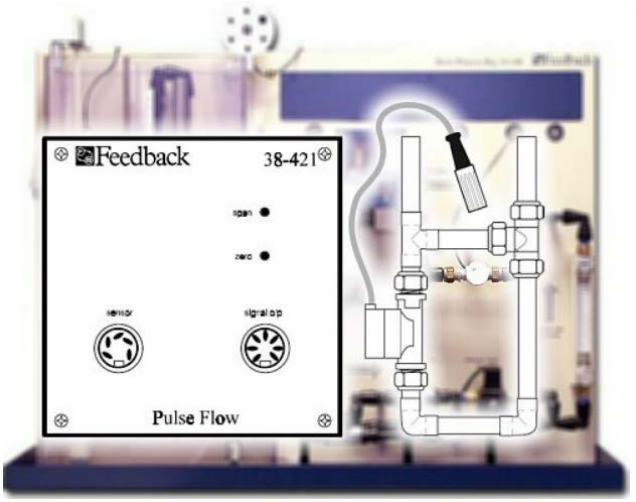


Imagen 92. Conjunto para la medición de caudal Feedback 38-420

1.5. Módulo de display digital p/4-20 mA Feedback 38-490

Se inserta en el camino de las señales 4-20 mA de cualquiera de los sensores o actuadores compatibles de la serie *Procon 38-xxx* de *Feedback*.

Muestra en un display LCD uno de los dos valores siguientes:

- 4 a 20 mA.
- 0 a 100%

Como pasa con los conjuntos anteriores, se alimenta de la interfaz de procesos *38-200*. Su base tiene una montura magnética para que sea capaz de adherirse a la pared de las maquetas gracias a un imán.

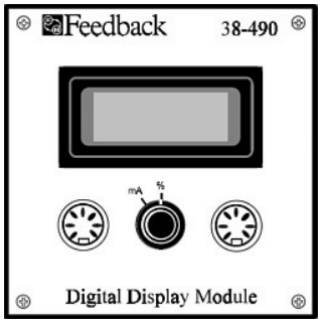


Imagen 93. Módulo de display digital Feedback 38-490

1.6. Planta de proceso p/temperatura Feedback 38-600

Imagen 94. Planta de proceso p/temperatura Feedback 38-600

ANEXO 2. Motor de corriente continua

Los motores de corriente continua se caracterizan por su versatilidad. Por medio de diversas combinaciones de devanados en derivación (*shunt*), en serie o en excitación separada de los campos, se puede conseguir que muestren una gran variedad de curvas características voltio-amperio y velocidad-par, tanto para el funcionamiento dinámico como para el estacionario. Debido a la facilidad de controlar el comportamiento de estos motores, se suelen usar en aplicaciones donde es necesaria una amplia gama de velocidades de motor o de control de la potencia de éste.

En los últimos años la tecnología de sistemas de control de estado sólido se ha desarrollado lo suficiente como para la creación de controladores para máquinas de corriente alterna (ca). Gracias a esto se comienzan a usar este tipo de sistemas en aplicaciones donde antes se asociaban exclusivamente con las máquinas de corriente continua. Sin embargo, los sistemas con motores de corriente continua seguirán aplicándose debido a su flexibilidad y a la sencillez relativa de sus lazos de control en comparación con los de corriente alterna.

Los principios fundamentales del funcionamiento de las máquinas de corriente continua son muy sencillos, pero suelen quedar eclipsados por la complejidad de la construcción de máquinas reales.

1. Características y componentes de la máquina de corriente continua

En la siguiente imagen aparecen esquemáticamente las características esenciales de una máquina de corriente continua. El estátor tiene los polos salientes y se excita mediante uno o más devanados de campo. La distribución de flujo en el entrehierro que crean los propios devanados de campo es simétrica respecto a la línea de dentro de los polos de campo. El rotor sustenta un conjunto de bobinas que giran con él y que se encargan de generar un campo magnético en cuadratura y, por consiguiente, generar el par de giro. El colector, que consiste a una especie de rectificador mecánico, se encarga de alimentar a cada bobina en el momento adecuado, con el fin de conservar la cuadratura de los campos.

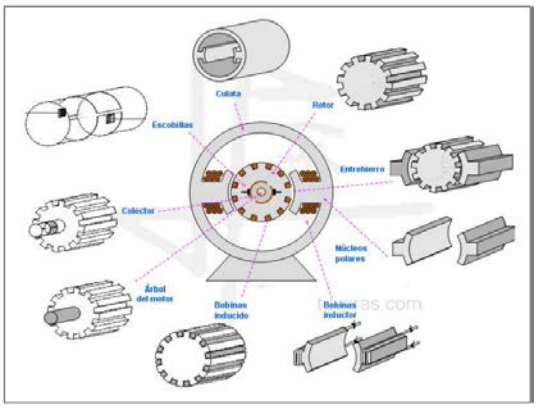


Imagen 95. Partes del motor de corriente continua

2. Modelo matemático de un motor de corriente continua

La siguiente figura muestra el modelo eléctrico del motor de corriente continua, de este motor podemos estudiar las ecuaciones base que describen el comportamiento de la máquina, pudiéndose obtener distintas curvas características.

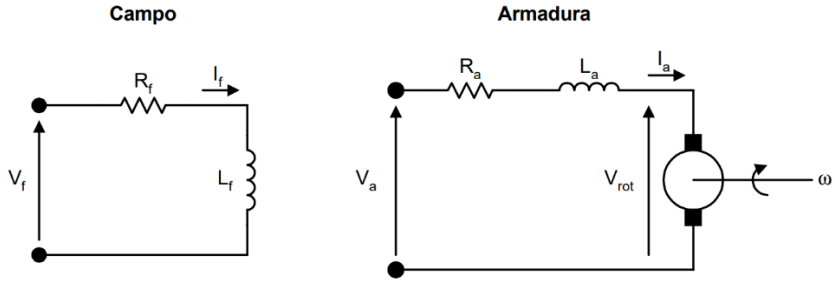


Imagen 96. Modelo eléctrico del motor de corriente continua

Las ecuaciones del campo se rigen por un sistema de primer orden (2.1), al igual que sucede en el rotor (2.2). Las ecuaciones magnéticas mecánicas relacionan el enlace entre el campo y la armadura (2.3) y la transferencia de energía hacia la carga (2.4 y 2.5).

$$V_f = R_f \cdot I_f + L_f \frac{dI_f}{dt} \quad (2.1)$$

$$V_a = R_a \cdot I_a + L_a \frac{dI_a}{dt} + Vrot \quad (2.2)$$

$$V_{rot} = G_{fq} \cdot I_f \cdot \omega \quad (2.3)$$

$$T = G_{fq} \cdot I_f \cdot I_a \quad (2.4)$$

$$T_{el} - T_{carga} = J \frac{d\omega}{dt} + D\omega \quad (2.5)$$

Donde:

V_f: voltaje de excitación de campo R_f: resistencia del devanado de campo I_f: corriente de campo L_f: inductancia de campo V_a: voltaje de armadura R_a: resistencia del devanado de armadura I_a: corriente de armadura L_a: inductancia de armadura V_{rot}: voltaje de reacción de armadura G_{fq}: constante de relación de enlace magnético entre el estátor y el rotor θ: posición angular ω: velocidad angular de rotación [rad/seg] T: par eléctrico T_{carga}: par de carga J: momento de inercia del rotor D: constante de roce

La máquina de corriente continua se puede conectar de diversas maneras, para empezar con el estudio de los lazos de control, se considerará un campo constante, es decir, V_f=cte. Esto genera una corriente de campo constante y se tiene que $G_{fq} \cdot I_f = K$. En estado estacionario las derivadas se hacen 0, por lo que las ecuaciones se reducen a las siguientes expresiones.

Conociendo lo anterior y tomando como referencia la siguiente imagen, que muestra el modelo eléctrico del motor de corriente continua, se realiza el cálculo para el estado estacionario. De este modelo se pueden deducir las ecuaciones base que describen el comportamiento de la máquina, pudiéndose obtener distintas curvas características.

El modelo matemático describe el comportamiento de un motor de corriente continua. Para estudiar el modelo físico del motor, tomamos una versión simplificada de su funcionamiento. La siguiente imagen representa un circuito eléctrico equivalente del estátor y el diagrama de cuerpo libre del rotor.

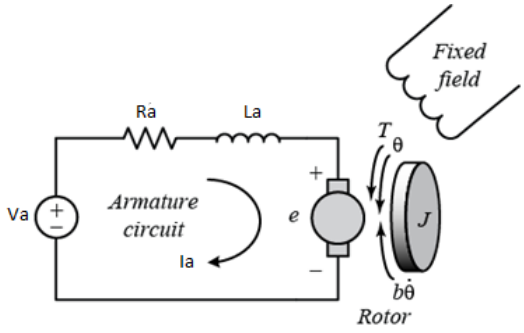


Imagen 97. Esquema de un motor de corriente continua

Se considera que el voltaje (V_a) de la fuente conectada al estátor del motor es la entrada, y que la velocidad rotacional del eje ($\dot{\theta}$) es la salida. Se asume que el rotor y el eje son partes rígidas. Si se va más en profundidad, se considera un modelo de fricción viscosa, esto es, el par de fricción es proporcional a la velocidad angular del eje. Las letras no presentadas en el anterior cuadro hacen referencia a los siguientes parámetros.

B: constante de fricción viscosa del motor
K_e: constante de fuerza electromotriz
K_t : constante del par motor

El devanado del inducido es una de las partes más importantes del motor, consiste en un enrollamiento de varias espiras que puede girar por la acción de un campo magnético constante. Dicho campo magnético puede ser generado por un imán o por un devanado de excitación situado en una bobina por la que circula una corriente de excitación $i_f(t)$, constante si queremos que el campo lo sea también. Al circular una corriente $i(t)$ por el devanado del inducido, originada de la interacción con el campo magnético, se ejerce sobre él un par $T(t)$ que es directamente proporcional al campo magnético y a la propia corriente del inducido $i_a(t)$. Como se supone que el campo magnético es constante, el par motor será proporcional a la corriente del inducido (2.6).

$$T(t) = K_t \cdot i_a(t) \tag{2.6}$$

Por otro lado, el giro de las espiras del devanado del inducido en presencia del campo magnético produce en los bornes del propio devanado una caída de tensión o fuerza contra electromotriz $e(t)$, proporcional a su velocidad de giro por un factor constante Ke.

$$e(t) = K_e \cdot \dot{\theta}(t) \tag{2.7}$$

El devanado del inducido es un conductor con una resistencia R y una inductancia L sobre el que hay que considerar la fuerza contra electromotriz como una fuente dependiente de la velocidad de giro. La ecuación de la malla del inducido se presenta a continuación (2.8).

$$v_a(t) = R_a \cdot i_a(t) + L_a \frac{di_a(t)}{dt} + K_e \cdot \dot{\theta}(t) \tag{2.8}$$

El par mecánico T(t) originado por el motor se emplea para introducir una velocidad angular a la carga y para vencer la fuerza de fricción. En unidades del SI, las constantes del par motor y de la fuerza contra electromotriz son iguales ($K_t = K_e$), por lo tanto, se usará K para referirse a ambas constantes.

$$T(t) = J \cdot \ddot{\theta}(t) + b \cdot \dot{\theta}(t) \tag{2.9}$$

Ahora, para obtener la función de transferencia, aplicamos la transformada de Laplace a las ecuaciones explicadas anteriormente para así poderlas expresar en términos de la variable s.

$$s(Js + b) \cdot \theta(s) = K \cdot I_a(s) \tag{2.10}$$

$$(L_a s + R_a) \cdot I_a(s) = V_a(s) - K \cdot s \cdot \theta(s) \tag{2.11}$$

La siguiente ecuación corresponde con la función de transferencia del sistema en lazo abierto, llegamos a ella despejando $I_a(s)$ de ambas ecuaciones e igualándolas. La velocidad rotacional es considerada como salida y el voltaje del inducido como entrada.

$$I_a(s) = \frac{V_a(s) - K \cdot s \cdot \theta(s)}{L_a s + R_a} \tag{2.12}$$

$$I_a(s) = \frac{s(Js+b) \cdot \theta(s)}{K} \tag{2.13}$$

E igualando las dos ecuaciones llegamos al siguiente paso (2.14).

$$\frac{s(Js+b) \cdot \theta(s)}{K} = \frac{V_a(s) - K \cdot s \cdot \theta(s)}{L_a s + R_a} \tag{2.14}$$

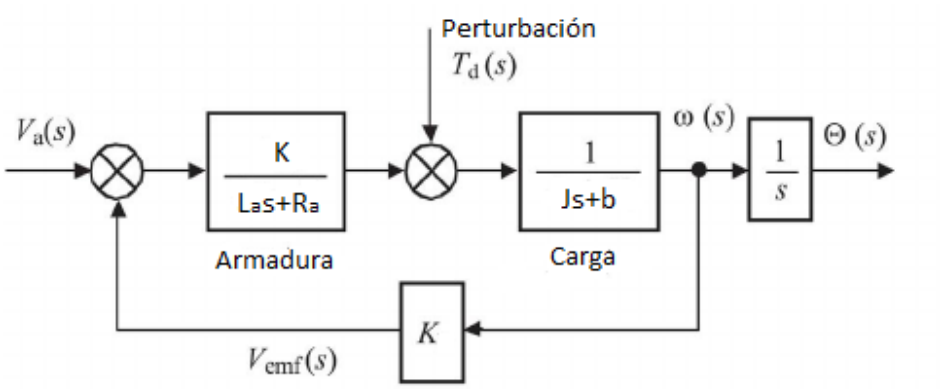


Imagen 98. Lazo cerrado motor corriente continua

Si hablamos de posición, la función de transferencia es:

$$G_{posición}(s) = \frac{\theta(s)}{V(s)} = \frac{K}{s[(R_a + L_a s) \cdot (Js + b) + K^2]} \tag{2.15}$$

Y, si queremos calcular la función de transferencia teniendo la velocidad como salida:

$$G_{velocidad}(s) = \frac{\dot{\theta}(s)}{V(s)} = \frac{K}{(R_a + L_a s) \cdot (Js + b) + K^2} \tag{2.16}$$

ANEXO 3. Válvulas de control

1. Definición

En el control automático de los procesos industriales la válvula de control tiene un papel muy importante en el bucle de regulación. Se trata de dispositivos mecánicos que controlan, retienen, regulan o dan paso a cualquier fluido entubado, es decir, su función es variar el caudal del fluido de control. Esta acción modifica también la variable medida, comportándose como un orificio de área continuamente variable. Su importancia en el bucle de control es tanta o más que el elemento primario, el transmisor y el controlador.

En automatismos, una válvula de control realiza la función de mantener dentro de un rango determinado (al abrirse o cerrarse) variables como: nivel, presión, caudal, etc. Dependiendo de lo que requiera un determinado proceso.

En el control de nivel de líquidos las válvulas se usan para mantener el nivel dentro de un rango apropiado, mediante la acción de abierto o cerrado de la válvula de control.

2. Generalidades

El funcionamiento de un sistema de tuberías depende de una buena elección y de la situación de las válvulas que regulan y controlan el tránsito de los fluidos en la instalación. Estas válvulas deben colocarse en lugares donde manejarlas sea fácil, de manera que se pueda conservar de manera adecuada la instalación sin llegar a interrumpir el funcionamiento de otros dispositivos conectados.

Para el diseño de una válvula debe tenerse en cuenta que:

- Sus deformaciones provocadas por las variaciones de temperatura y presión, así como las dilataciones de las tuberías conectadas, no deterioren ni deformen el asiento.
- El vástago y el collarín del prensaestopas deben permitir colocar la empaquetadora fácilmente y con rapidez, y los discos y los asientos deben diseñarse y construirse con materiales aptos para que la válvula siga cerrando bien durante un período razonable de servicio activo.

Tras las bombas y los motores, las válvulas son los componentes más importantes de los circuitos hidráulicos. Con el empleo adecuado de los diferentes tipos de válvulas se consigue llevar a cabo operaciones de control múltiples, complejas y automáticas. Las válvulas pueden servir para realizar tres funciones diferentes:

- Controlar la presión: limitar la presión del circuito para protegerlo o para reducir la fuerza aplicada o el par ejercido por el cilindro o por un motor rotativo: las válvulas limitan la presión en una rama del circuito a un valor inferior a la presión de trabajo del circuito principal: se hacen cargo de la sucesión de operaciones que se llevan a cabo entre dos ramas de un circuito.

- Controlar el caudal: regulan, por ejemplo, la velocidad con la que un cilindro hidráulico se mueve.
- Controlar la dirección: impiden el paso del fluido en un sentido, pero le dejan cruzar en el sentido contrario.

3. Partes internas de la válvula

Las partes internas de la válvula son las piezas metálicas desmontables que tienen un contacto directo con el fluido, estas piezas son el vástago, la empaquetadora, el collarín de lubricación en la empaquetadora, los anillos de guía del vástago, el obturador y el asiento o asientos. El obturador y el asiento constituyen el corazón de la válvula al controlar el caudal gracias al orificio de paso variable que forman al variar su posición relativa, y que además tienen la misión de no permitir el paso del fluido.

La elección del obturador y del asiento se basa en tres puntos principales:

- Empleo de materiales normales y especiales aptos para soportar la corrosión, la erosión y el desgaste provocados por el fluido.
- Características de caudal en función de la carrera.
- Tamaño normal o reducido que permite obtener varias capacidades de caudal de la válvula con el mismo tamaño del cuerpo.

4. Cuerpo de la válvula

El cuerpo de la válvula es construido con la finalidad de resistir la temperatura y la presión del fluido sin tener ninguna perdida, su tamaño debe de ser adecuado para el caudal que controle y debe ser resistente a la erosión y corrosión producidas por el fluido.

Tanto el cuerpo como las conexiones a la tubería están normalizados de acuerdo con las presiones y temperaturas de trabajo en las normas DIN y ANSI, entre otras.

En este apartado cabe resaltar tres puntos:

- Las conexiones roscadas se utilizan hasta unos 50 mm.
- Las bridas utilizadas pueden ser planas, con resaltes, machihembradas, machihembradas con junta de anillo.
- Las conexiones pueden ser con encaje o con soldadura a tope, las primeras se emplean para válvulas de hasta 50 mm, y las segundas desde unos 62 mm a tamaños mayores.

El cuerpo suele ser de hierro, acero y acero inoxidable, en algunos casos también pueden ser construidos de monel, hastelloy B o C, etc.

5. Tapa de la válvula

La tapa de la válvula tiene el propósito de unir el cuerpo con el accionador, para que el fluido no se escape por la tapa es necesario tener una empaquetadora entre dicha tapa y el vástago. Lo ideal es que la empaquetadura sea elástica, con un coeficiente bajo de rozamiento, ser químicamente inerte y aislante eléctrico con el fin de no formar un puente galvánico con el vástago, corroyendo partes de la válvula. La empaquetadura que normalmente se utiliza está hecha de teflón, teniendo así una temperatura máxima de servicio de 220ºC. Si supera esta temperatura o incluso si no llega a alcanzarse es necesario utilizar otro material o alejar la empaquetadura del cuerpo de la válvula para establecerse así un gradiente de temperaturas entre el fluido y la estopada y permitir así que esta última puede trabajar satisfactoriamente.

Hay que saber que la empaquetadura no realiza un sello perfecto para el fluido, en el caso de sustancias corrosivas, tóxicas, radioactivas o muy valiosas hay que asegurar un cierre total en la estopada.

6. Materiales

El obturador y los asientos se suelen fabricar con acero inoxidable ya que este material es muy resistente a la corrosión y a la erosión del fluido. Cuando la erosión del fluido es baja, puede usarse PVC, fluorocarbonos y otros materiales blandos, solos o reforzados con fibras de vidrio o grafito. En algunas válvulas se pueden usar obturadores y asientos de cerámica.

7. Golpe de ariete

Cuando en una columna de fluido, que está en movimiento, se reduce bruscamente su velocidad o se detiene debido a, por ejemplo, el cierre rápido de una válvula, la presión interna aumenta considerablemente por la cantidad de movimiento del fluido, produciendo de esta manera una pulsación.

A menudo el ruido que provoca es peculiar, es como el del golpe de un martillo en la tubería, de aquí el término “golpe de ariete”. Esta pulsación hace que la cañería deje de ser estable, ya que los diversos esfuerzos provocados pueden llegar a romperla. La viscosidad del fluido es un factor del que depende también el golpe de ariete, mientras menos viscoso sea, menor es el golpe y viceversa.

8. Característica de caudal inherente

La característica de caudal de la válvula es la relación que existe entre la posición del obturador y el caudal de paso del fluido. Esta relación la determina el obturador, las variaciones características se obtienen mecanizando el obturador para que, al modificar la cerrera del orificio de paso variable situado entre el contorno del obturador y el asiento, configure la característica de la válvula.

Con un obturador con característica de apertura rápida, el caudal aumenta mucho al principio de la carrera, llegando rápidamente al máximo. Con un obturador de característica lineal, el caudal es directamente proporcional a la carrera, mientras que teniendo un obturador con característica isoporcentual cada incremento de carrera del obturador produce una variación de caudal proporcional al caudal que fluía antes de la variación. La variación de caudal es pequeña al principio de la carrera de la válvula, y al final, pequeños incrementos en la carrera provocan grandes variaciones de caudal.

9. Corrosión y erosión en las válvulas

Ningún material puede resistir la corrosión de todos los fluidos, motivo por el cual en muchos casos es necesario utilizar materiales combinados, que se elegirán dependiendo del medio específico donde vayan a trabajar. Cuando el material es caro o no adecuado, pueden utilizarse materiales de revestimiento como plásticos, elastómeros, fluorocarbonos, vidrio, plomo y tantalio.

La erosión es producida cuando las partículas del fluido chocan con la superficie de la válvula, este fenómeno obliga a seleccionar el tipo y material del cuerpo y del obturador

con el fin de resistirla, particularmente en condiciones extremas de presión diferencial y de temperatura.

10. Tipos de válvulas

Muchas son las maneras de clasificar las válvulas, una de ellas es:

- Válvulas de apertura lenta: formadas por un vástago de tornillo, que debe ser girado varias veces para abrir la compuerta totalmente, permiten un tiempo de llenado de la cañería. Los ejemplos más comunes de este tipo de válvulas son esclusa, globo, Saunders, de aguja y de diafragma.

- Válvulas de apertura rápida: son aquellas en las que un solo golpe puede provocar el cambio de sentido completo. Generalmente casi todas estas válvulas pasan de totalmente cerradas a totalmente abiertas en un cuarto de vuelta (90º). Las más usadas son macho tapón lubricado, de mariposa, de bola y de retención.

En el mercado existe una gran variedad en el mercado, las más comunes son: válvulas manuales y automáticas. Dentro de las válvulas manuales podemos encontrarnos con:

10.1. Válvulas de tipo compuerta

Son utilizadas para el flujo de fluido limpios y sin interrupción, este tipo de válvulas no son aconsejables para estrangulamiento debido a que posee un disco que se mueve dentro del cuerpo de la válvula, entonces el estrangulamiento causaría una erosión, arruinando su funcionamiento.

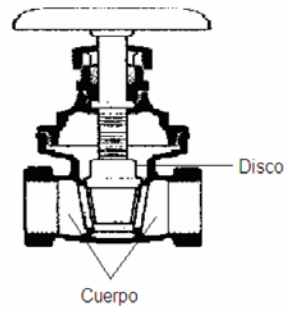


Imagen 99. Válvula de tipo compuerta

Existen multitud de tipos de válvulas de compuerta, las cuales se diferencian por el tipo de disco para el cierre: válvula de compuerta tipo cuña sólida, tipo flexible, tipo abierta, de cierre rápido, de guillotina.

Normalmente este tipo de válvulas son construidas con cuerpo de bronce, hierro, latón acero fundido. Mientras que en su interior son normalmente de bronce, acero inoxidable, acero aleado, estelita, cromo o molibdeno.

10.2. Válvulas de retención o check

Estas válvulas se usan como medida de seguridad, para evitar que el flujo retroceda dentro de la tubería, también se usan para mantener llena la tubería cuando la bomba sufre no funciona automáticamente.

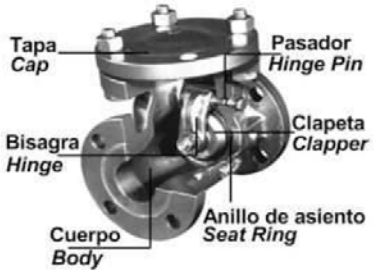


Imagen 100. Válvula de retención o Check

Este tipo de válvula se usa en serie con las de compuerta y pueden funcionar tanto en posición horizontal como en vertical. En este dispositivo la presión del fluido circulante abre la válvula, mientras que el peso del mecanismo de retención y cualquier inversión en el flujo la cierra.

Varios son los tipos de válvulas de retención que existen, y su selección depende de la temperatura, de la caída de presión que producen y la limpieza de fluido.

Algunas válvulas de retención se pueden equipar con pesos externos, esto producirá el cierre rápido del disco. Este tipo de válvula se compone principalmente de asiento, cuerpo, disco y pasador oscilante.

10.3. Válvulas de macho

Consiste en una válvula de un cuarto de vuelta. Dado que el flujo que pasa por la válvula es suave y sin interrupción, existen pocas turbulencias dentro de ella y, por lo tanto, la caída de presión es baja.

El macho es una parte cónica y cilíndrica y tiene un conducto por donde fluye el líquido. En la posición abierta, la cavidad en el macho conecta los extremos de entrada y salida de la válvula y permite el flujo lineal.

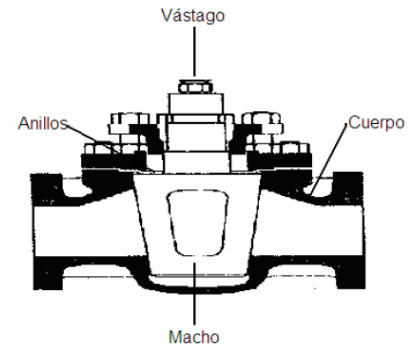


Imagen 101. Válvula de macho

Este tipo de válvulas tienen un pequeño problema, y es que se endurecen si no son accionadas periódicamente. Aunque tienen la cualidad de adaptarse fácilmente al tipo de orificios múltiples.

Las válvulas macho se utilizan en plantas con procesos químicos. Además, con este tipo de válvulas conseguimos rebajar el costo de nuestra instalación ya que su precio es relativamente bajo en comparación con otras válvulas.

10.4. Válvulas de bola

Estas válvulas tienen un macho esférico que controla el movimiento del líquido, consisten en válvulas macho modificadas y tienen un uso limitado debido al asentamiento de metal con metal, que no permite el cierre.

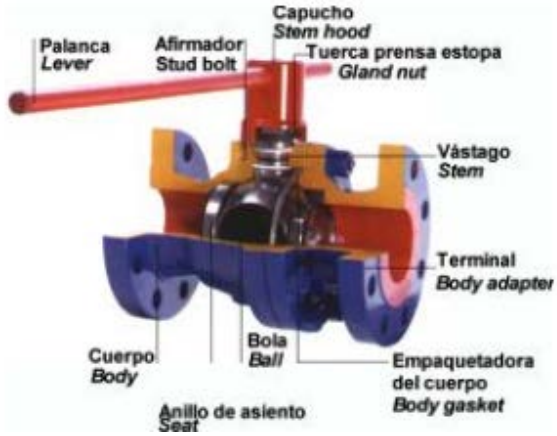


Imagen 102. Válvula de bola

Debido a los avances en la fabricación de plásticos, los asientos metálicos se han cambiado por plastómeros modernos.

La válvula de bola está formada por un cuerpo con orificio de Venturi y anillos de asientos, una bola para provocar el cierre y una jaula con vástago para desplazar la bola en función del orificio. Son rápidas de operar, su mantenimiento es fácil y su caída de presión es función del tamaño del orificio.

Esta válvula está limitada a las temperaturas y presiones que permite el material del asiento. Se puede emplear para vapor, aceite, agua, gas, fluidos corrosivos, pastas aguadas y materiales pulverizados secos.

10.5. Válvulas de mariposa

Son válvulas de un cuarto de vuelta, su nombre es debido a la acción tipo aleta del disco regulador de flujo, el que opera en torno a un eje que está en ángulo recto al flujo.

Consiste en un disco (también llamado hoja o chapaleta), un cuerpo con cojinetes y empaquetadora para sellamiento y soporte, y un eje.

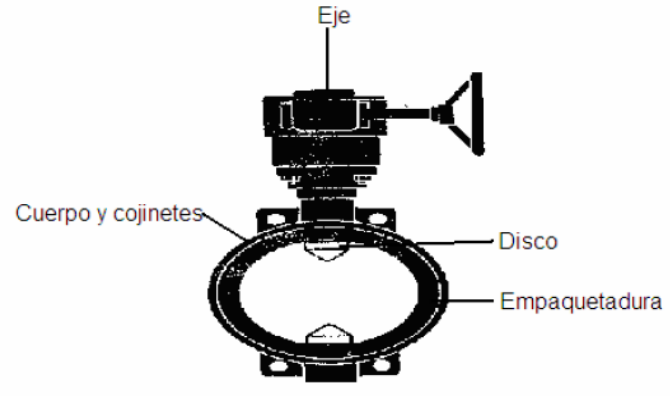


Imagen 103. Válvula de mariposa

Su uso es recomendado en trabajos donde el fluido contiene gran cantidad de sustancias sólidas en suspensión, ya que por su forma es difícil que dichos sólidos se acumulen en su interior y entorpezcan su funcionamiento.

Aunque sean excelentes para el control del fluido, es común utilizarlas en el servicio de corte y estrangulamiento cuando se trabaja con grandes volúmenes de gases y líquidos a presiones relativamente bajas. Para la estrangulación el disco se mueve a una posición intermedia, en el cual se mantiene por medio de un seguro. Para el corte, el disco obstruye totalmente el paso del fluido.

10.6. Válvulas de diafragma

Este tipo de válvulas son de vueltas múltiples, se utilizan principalmente para el corte y la estrangulación de líquidos con gran cantidad de sólidos en suspensión, además desempeñan una serie de servicios importantes para el control de fluido.

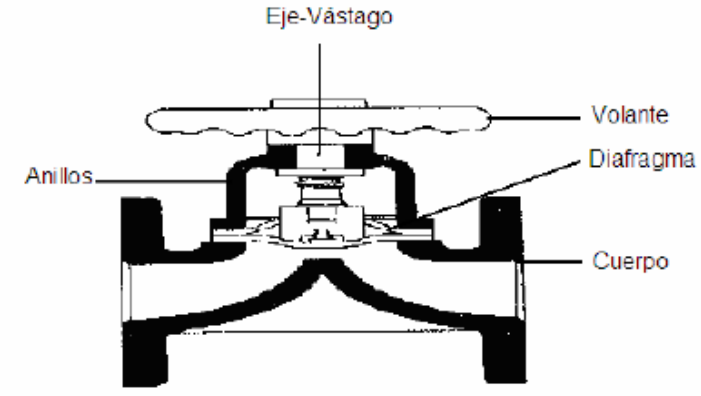


Imagen 104. Válvula de diafragma

Sus aplicaciones son:

Para presiones bajas y pastas aguadas que corroerían y obstruirían a la mayoría del resto de equipos.

- Para fluidos corrosivos, materiales pegajosos o viscosos.
- Etc.

Cuando se abre la válvula, se produce la elevación del diafragma, quedando éste fuera de la trayectoria de flujo, de esta manera el líquido tiene un paso suave y sin obstrucciones. Cuando la válvula se cierra, el diafragma se asienta con rigidez contra un vertedero o zona circular en el fondo de la válvula.

10.7. Válvulas de globo

Su principal función es regular el flujo del fluido. Estas válvulas controlan el fluido desde el goteo hasta el sellado hermético. Además, siguen siendo eficientes para cualquier posición del vástago.

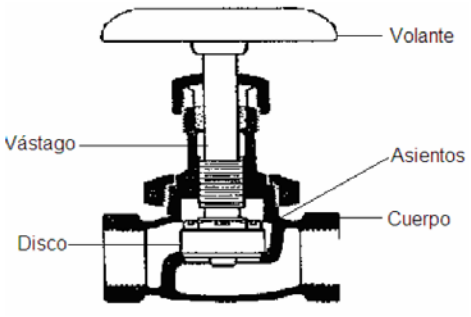


Imagen 105. Válvula de Globo

Debido a una caída de tensión fuerte (siempre controlada) se utilizan en servicios donde la válvula de compuerta no funciona adecuadamente.

Estas válvulas ocupan el mismo espacio y pesan casi igual que las válvulas de compuerta. La construcción interna es algo que las caracteriza, debido a que poseen un disco o macho cuyo movimiento se da en forma perpendicular al fluido dentro del cuerpo de la válvula.

Sus partes principales son volante, vástago, bonete, asientos, disco y cuerpo.

10.8. Válvula de aguja

La válvula de aguja es un tipo especial de la válvula de globo con macho. Se trata de una aguja delgada y cónica que se apoya en un orificio pequeño de diferente conicidad. El cuerpo puede ser convencional de globo o en ángulo. Muchas de las características de diseño son similares a las de la válvula de globo.

Por lo general, se utilizan como válvulas para instrumentos o en sistemas hidráulicos, aunque no para altas temperaturas.

Imagen 106. Válvula de aguja

La principal ventaja que presenta esta válvula es que está especialmente adaptada para un control muy preciso de bajos volúmenes de flujo. Como desventaja, y siendo un factor a tener en cuenta, es que la caída de presión es considerable.

El módulo *FEEDBACK* de nivel y flujo *PROCON 38-001* cuenta con este tipo de válvula de aguja y es la servoválvula con la que se trabaja durante este proyecto.

10.9. Válvulas automáticas de control

Este elemento suele constituir el último paso de un lazo de control situado en la línea de proceso y su comportamiento se basa en un orificio cuya sección de paso varía continuamente con la finalidad de controlar un caudal de una forma determinada.

Las válvulas de control están formadas fundamentalmente por dos partes: la parte motriz y el cuerpo.

11. Actuador

El actuador también es conocido como accionador o motor, puede ser de diversas formas: neumático, eléctrico o hidráulico. Los neumáticos y lo eléctricos son los más utilizados por ser los más sencillos y los que tienen una acción más rápida. Alrededor del 90% de las válvulas utilizadas en la industria son accionadas neumáticamente. Los actuadores neumáticos están formados por un diafragma, un vástago y un resorte.

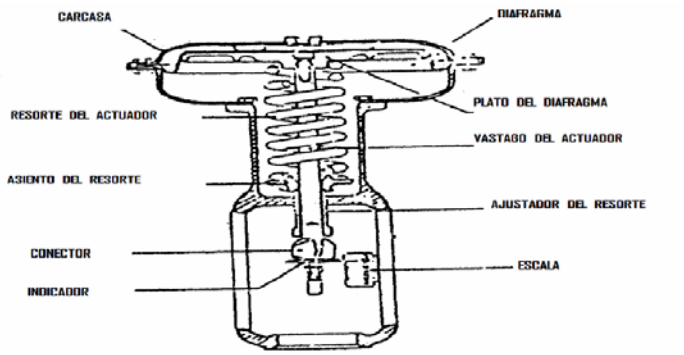


Imagen 107. Actuador de una válvula de control

12. Servoválvula

Una servoválvula es un tipo de válvula direccional hidraúlica de presión que está controlada eléctricamente. Son accionadores neumáticos o hidráulicos que conectan dos vías o más por las que circula un fluido. Lo que las diferencia de las válvulas normales es que éstas son del tipo todo o nada, mientras que las servoválvulas nos permiten controlar la presión o el caudal.

El control de paso del fluido es más preciso que en las válvulas convencionales ya que se obtiene por la variación gradual de la corriente eléctrica en un motor de par que actúa sobre un dispositivo llamado aleta. Esto, al mismo tiempo, controla de forma precisa el flujo y la presión del fluido que mueve el carrete, responsable de dirigir el fluido para las salidas de servicio de la válvula. De esta forma se asegura una gran precisión en el posicionamiento, velocidad, fuerza y presión del equipo que está siendo accionado.

Las servoválvulas son dispositivos utilizados principalmente en sistemas hidráulicos que requieren precisión en esta operación, como pueden ser los controles de vuelo de la aeronave y la robótica industrial.

Como se ha comentado, se utilizan para controlar la presión o el caudal. Se caracterizan por tener un estructura sencilla y robusta, el elemento empleado de junta puede ser una membrana, si las presiones en las que se trabaja son menores de 20 bares, ó un émbolo obturador, recomendado para presiones de aplicación de hasta 450 bares.

Mandar significa dar una orden con el fin de poner todos los medios para conseguir el objetivo, este objetivo no se consigue de manera controlada y, aunque no se obtenga con exactitud, lo medios no sufren correcciones. Si se habla de regular significa que, además de dar la orden para que los medios se pongan para conseguir un objetivo, se controla su consecución, si no se cumple se realizan las acciones necesarias para corregir los medios indicados para conseguir la meta propuesta.

Estos dispositivos disponen de realimentación interna, que es la encargada de la regulación anteriormente descrita, de esta manera se consigue un alto nivel de exactitud en los circuitos hidráulicos, superior incluso que con válvulas proporcionales y más elevado que con las convencionales.

ANEXO 4. Soporte ARDUINO de Simulink

En este anexo se detalla cómo funciona el paquete adicional de *Simulink* para trabajar con la plataforma *ARDUINO*.

El *Simulink Support Package* para el hardware *ARDUINO* permite crear y ejecutar modelos de *Simulink* en placas de *Arduino*. Incluye una biblioteca de bloques de *Simulink* para configurar y acceder a sensores, actuadores e interfaces de comunicación Arduino. También permite monitorizar y sintonizar de forma interactiva los algoritmos desarrollados en *Simulink* mientras se ejecutan en *Arduino*.

Las placas microprocesadoras que permite configurar este paquete son: Arduino UNO, Arduino MEGA 2560, Arduino MEGA ADK, Arduino NANO 3.0, Arduino PRO, Arduino FIO, Arduino MINI, Arduino LEONARDO, Arduino MICRO, Arduino ESPLORA, Arduino ROBOT CONTROL BOARD, Arduino ROBOT MOTOR BOARD, Arduino LILYPAD USB Y Arduino DUE.

Es posible desarrollar modelos y algoritmos en *Simulink*, simularlos para comprobar que se ejecuten según lo previsto y transferir el algoritmo completo en el dispositivo para su completa ejecución (a través de USB, vía Wi-Fi o conexión Ehternet). Existe también la posibilidad de poder ajustar parámetros desde el modelo de *Simulink* mientras el algoritmo se ejecuta en la placa.

Este paquete incluye:

- Una biblioteca de bloques *Simulink* que conectan a las placas ARDUINO con numerosos sensores de entrada y de salida.
- Una biblioteca de bloques *Ethernet Shield* que permiten realizar una comunicación con Arduino vía *Ethernet.*
- Una biblioteca de bloques *Wifi Shield* con las funciones necesarias para garantizar una conexión vía Wi-Fi con *Arduino*.

Para descargar este paquete se pueden seguir las instrucciones que se detallan en la página web oficial de *Matlab & Simulink*.

https://es.mathworks.com/help/supportpkg/arduino/setup-and-configuration.html

1. Librerías de bloques Arduino

Como se ha especificado anteriormente, existen 3 librerías para este paquete, una común que recoge todos los bloques necesarios para trabajar con entradas y salidas digitales y analógicas, así como su comunicación puerto serie. Y las otras dos hacer referencia a los bloques necesarios para implementar una comunicación vía Wi-Fi o Ethernet con la placa de *Arduino*.

1.1. Common

El conjunto *Common* recoge los siguientes bloques:

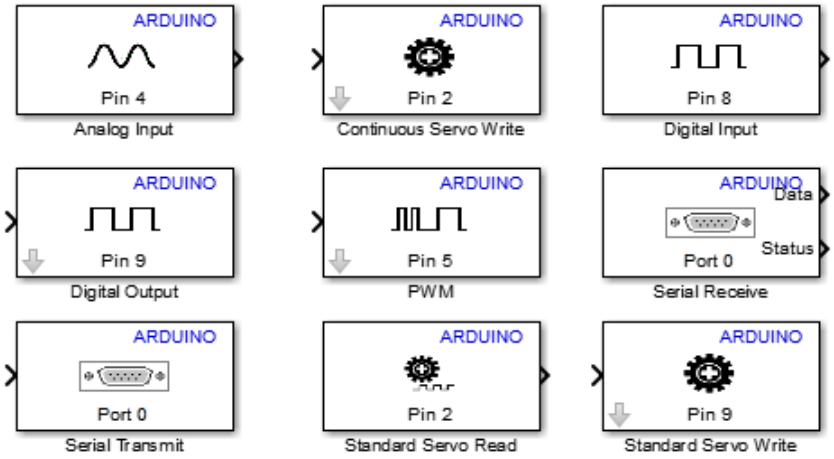


Imagen 108. Bloques del conjunto Common

El funcionamiento de cada uno se explica en la siguiente tabla.

Bloques del apartado *Common*	
Analog Input	Mide el voltaje de una entrada analógica
Continuous Servo Write	Establece la velocidad del eje del servomotor de rotación continua
Digital Input	Obtiene el valor lógico de una entrada digital
Digital Output	Establece el valor lógico de una salida digital
PWM	Genera la forma de onda cuadrada en un pin de salida analógica especificado
Serial Receive	Recibe una matriz de datos [Nx1] en el puerto serie especificado y la escribe en la salida del bloque de datos. Cuando los datos no están disponibles, escribe un 0 en la salida del bloque de datos.
Serial Transmit	Envía los datos almacenados en el puerto serie especificado
Standard Servo Read	Obtiene la posición del eje del servomotor estándar en grados
Standard Servo Write	Establece la posición del eje del servomotor estándar

Tabla 5. Librería Common Arduino

1.2. Ethernet Shield

Para realizar una comunicación *Ethernet* se recogen los siguientes bloques:

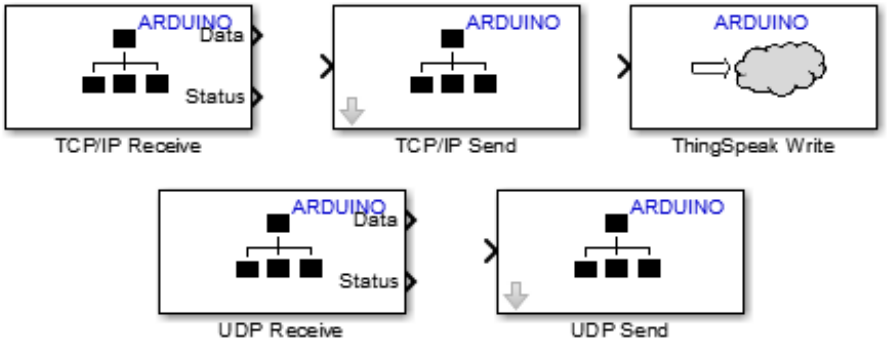


Imagen 109. Bloques del conjunto Ethernet Shield

El funcionamiento de cada uno se explica en la siguiente tabla.

Bloques del apartado Ethernet *Shield*	
TCP/IP Receive	Recibe el mensaje TCP/IP en una red cableada.
TCP/IP Send	Envía el menaje TCP/IP desde el servidor.
ThingSpeak Write	Lee los datos almacenados en un canal ThingSpeak.
UDP Receive	Recibe el mensaje UDP en la red cableada.
UDP Send	Envía el mensaje UDP a la interfaz remota.

Tabla 6. Librería Ethernet Shield Arduino

1.3. Wi-Fi Shield

Para la comunicación vía Wi-Fi para *Arduino*, *Matlab & Simulink* ofrece los siguientes bloques:

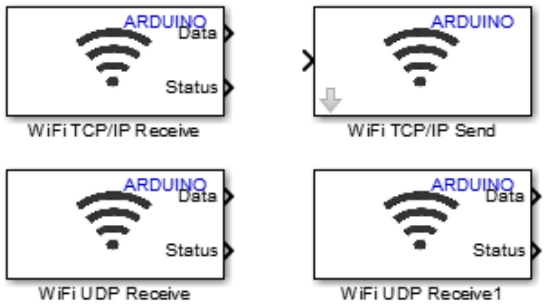


Imagen 110. Bloques del conjunto Wi-Fi Shield

El funcionamiento de cada uno de ellos se explica a continuación en la tabla.

Bloques del apartado *Wi-Fi Shield*	
WiFi TCP/IP Receive	Recibe datos a través de una red TCP/IP desde el host remoto en la red inalámbrica
Wifi TCP/IP Send	Envía datos a través de una red TCP/IP a un host remoto a través de una red inalámbrica.
Wifi UDP Receive	Recibe los datos del host UDP en la red inalámbrica.
Wifi UDP Send	Envía los datos al host UDP en la red inalámbrica.
Wifi ThingSpeak Read	Lee los datos almacenados en el canal ThingSpeak.
Wifi ThingSpeak Write	Publica datos en el Internet de las Cosas (IoT) usando WiFi ThingSpeak.

Tabla 7. Librería WiFi Shield Arduino

2. Comunicación con Arduino Hardware

Una vez montado el circuito *Arduino* para ejecutar el modelo de *Simulink,* es necesario configurar el modelo que se va a cargar en la placa. Para ello es necesario seguir los siguientes pasos.

1. En el modelo de *Simulink*, pulsar ***Tools*** > ***Run on Target Hardware*** > ***Prepare to Run…***
2. En la ventana ***Configuration Parameters***, configurar la pestaña ***Target Hardware*** como *Arduino UNO*.
3. Por último, en la pestaña ***Host To Target Connection*** se ponen los parámetros: ID del dispositivo y dirección IP el mismo, los cuales ya aparecen establecidos por defecto.

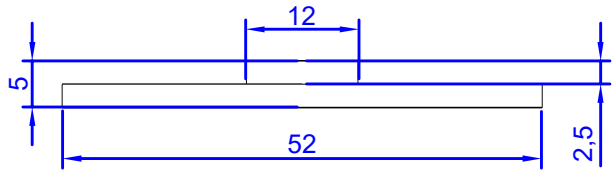
Configurado el modelo específico para Arduino, solo falta transferirlo a la placa pulsando ***Deploy To Hardware*** en ***Modo External***.

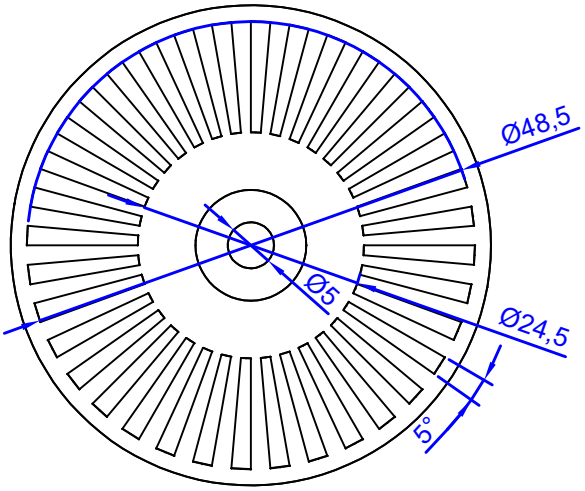
"DISEÑO, DESARROLLO Y CONSTRUCCIÓN DE SISTEMA DE CONTROL PARA SERVOVÁLVULA"

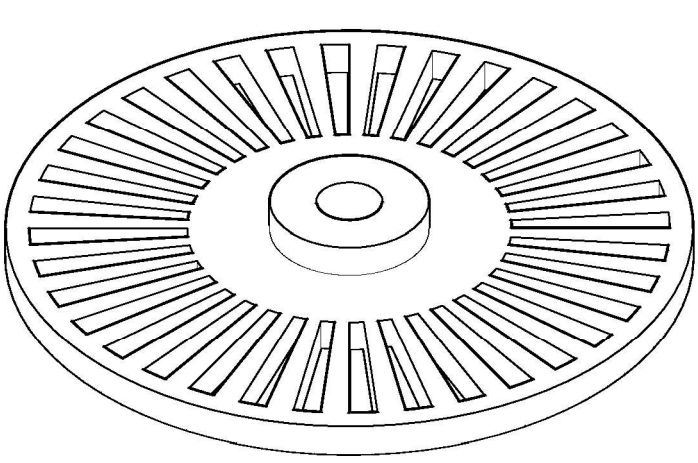
DOCUMENTO Nº 4 PLANOS

Peticionario:	Universidad de La Rioja
Informantes:	Miguel de Gregorio Santamarina
	Estudiante de Ingeniería Electrónica Industrial y Automática

ÍNDICE



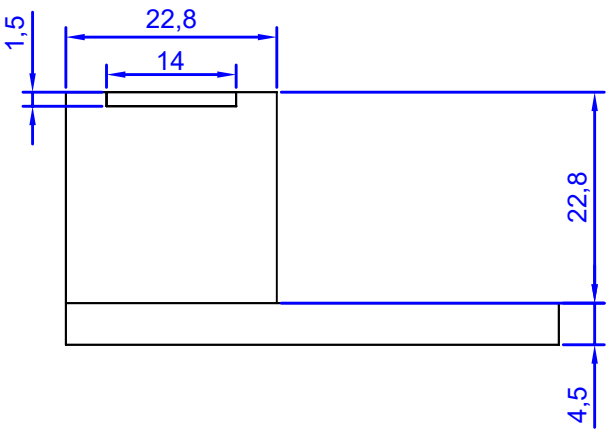


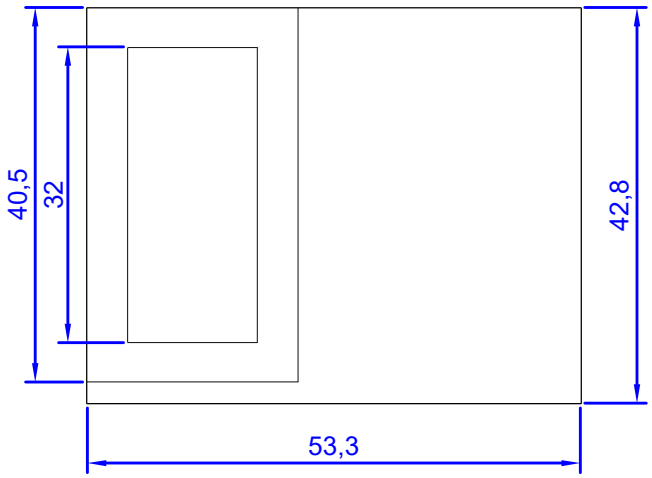


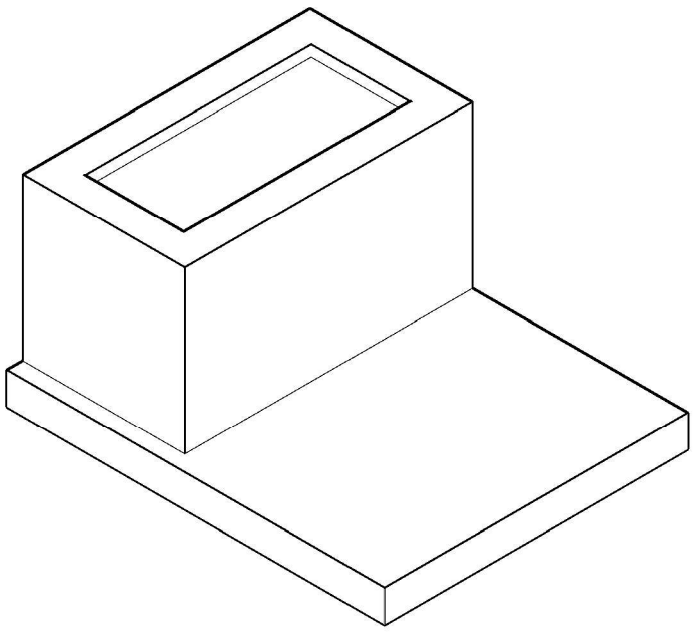
	FECHA	NOMBRE		ESCUELA TÉCNICA SUPERIOR DE INGENIERÍA INDUSTRIAL *Universidad de La Rioja* Grado en Ingeniería Electrónica Industrial y Automática	
Dibujado	28/05/2018	Miguel de Gregorio Santamarina			
Comprobado					
Dib. S. Norma	U.N.E.	Tolerancia general			

Escalas:	**TRABAJO FIN DE GRADO**	**NÚMERO: 1**
PROYECCIÓN	**RUEDA PARA ENCODER ÓPTICO**	

CREADO CON UNA VERSION PARA ESTUDIANTES DE AUTODESK



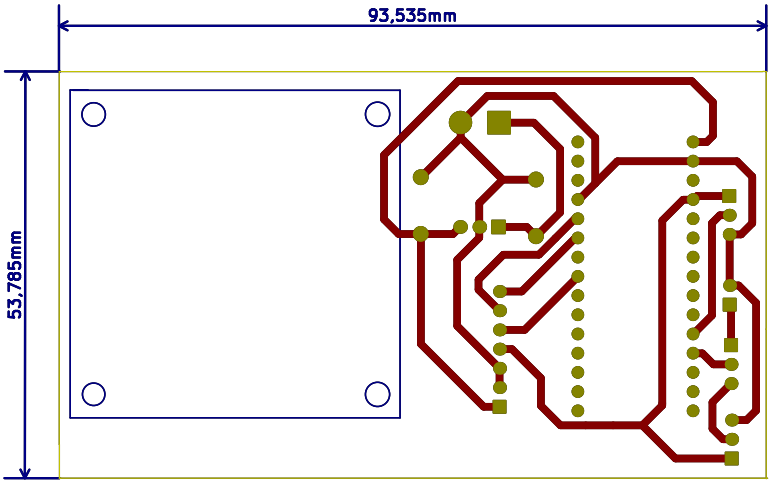




	FECHA	NOMBRE		ESCUELA TÉCNICA SUPERIOR DE INGENIERÍA INDUSTRIAL *Universidad de La Rioja* Grado en Ingeniería Electrónica Industrial y Automática	
Dibujado	28/05/2018	Miguel de Gregorio Santamarina			
Comprobado					
Dib. S. Norma	U.N.E.	Tolerancia general			

Escalas:	**TRABAJO FIN DE GRADO**	**NÚMERO: 2**
PROYECCIÓN	**SOPORTE PARA ENCODER ÓPTICO**	

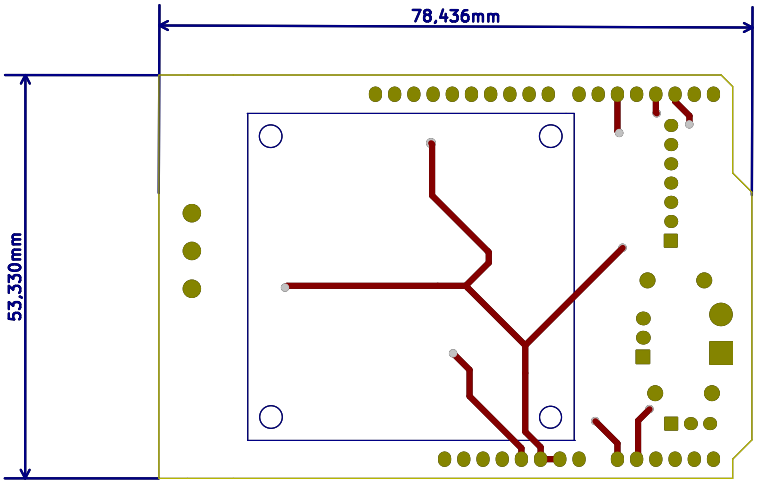
CREADO CON UNA VERSION PARA ESTUDIANTES DE AUTODESK
CREADO CON UNA VERSION PARA ESTUDIANTES DE AUTODESK

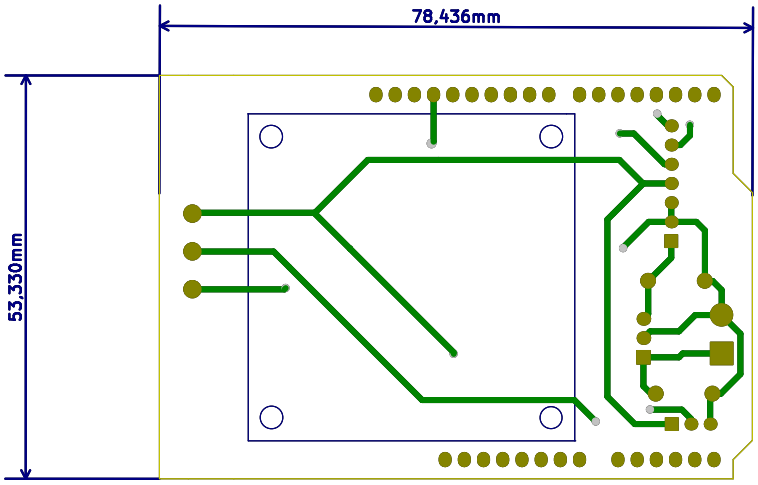


	FECHA	NOMBRE		ESCUELA TÉCNICA SUPERIOR DE INGENIERÍA INDUSTRIAL *Universidad de La Rioja* Grado en Ingeniería Electrónica Industrial y Automática	
Dibujado	04/07/2018	Miguel de Gregorio Santamarina			
Comprobado					
Dib. S. Norma	U.N.E.	Tolerancia general			

Escalas:	**TRABAJO FIN DE GRADO**	**NÚMERO: 3**
PROYECCIÓN	**PCB DISEÑADA PARA ARDUINO NANO**	

CREADO CON UNA VERSION PARA ESTUDIANTES DE AUTODESK
CREADO CON UNA VERSION PARA ESTUDIANTES DE AUTODESK
CREADO CON UNA VERSION PARA ESTUDIANTES DE AUTODESK

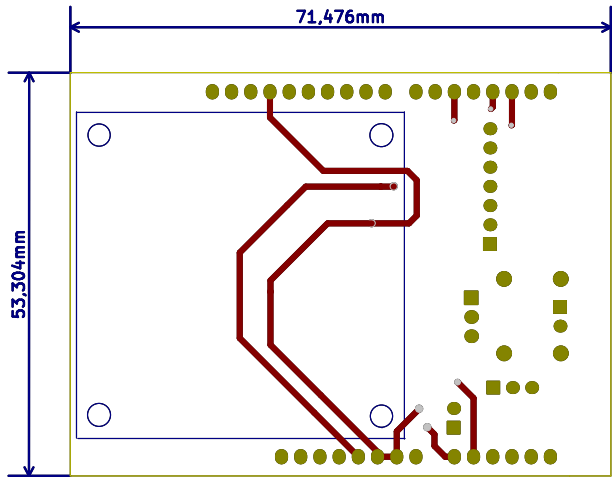


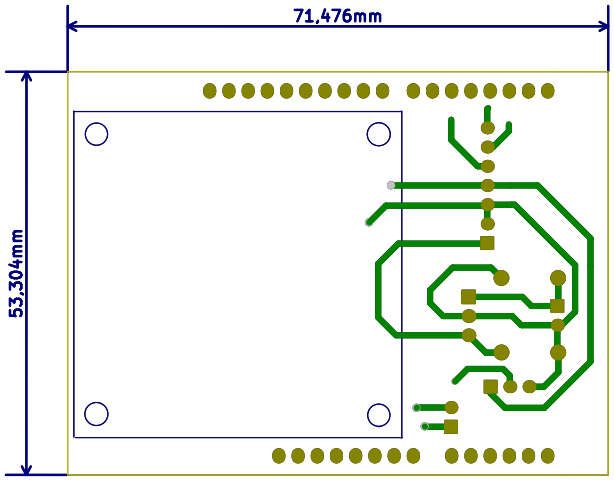


	FECHA	NOMBRE		ESCUELA TÉCNICA SUPERIOR DE INGENIERÍA INDUSTRIAL *Universidad de La Rioja* Grado en Ingeniería Electrónica Industrial y Automática	
Dibujado	04/07/2018	Miguel de Gregorio Santamarina			
Comprobado					
Dib. S. Norma	U.N.E.	Tolerancia general			

Escalas:	TRABAJO FIN DE GRADO	NÚMERO: 4
PROYECCIÓN	PRIMERA PCB DISEÑADA PARA ARDUINO UNO	

CREADO CON UNA VERSION PARA ESTUDIANTES DE AUTODESK

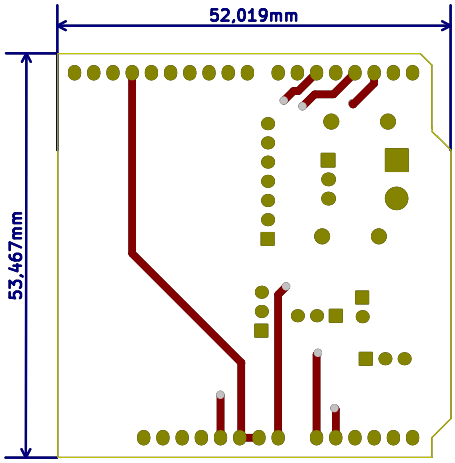


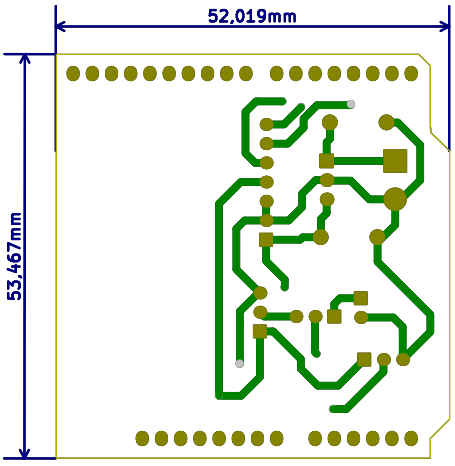


	FECHA	NOMBRE		ESCUELA TÉCNICA SUPERIOR DE INGENIERÍA INDUSTRIAL *Universidad de La Rioja* Grado en Ingeniería Electrónica Industrial y Automática	
Dibujado	04/07/2018	Miguel de Gregorio Santamarina			
Comprobado					
Dib. S. Norma	U.N.E.	Tolerancia general			

Escalas:	**TRABAJO FIN DE GRADO**	**NÚMERO: 5**
PROYECCIÓN	**SEGUNDA PCB DISEÑADA PARA ARDUINO UNO**	

CREADO CON UNA VERSION PARA ESTUDIANTES DE AUTODESK

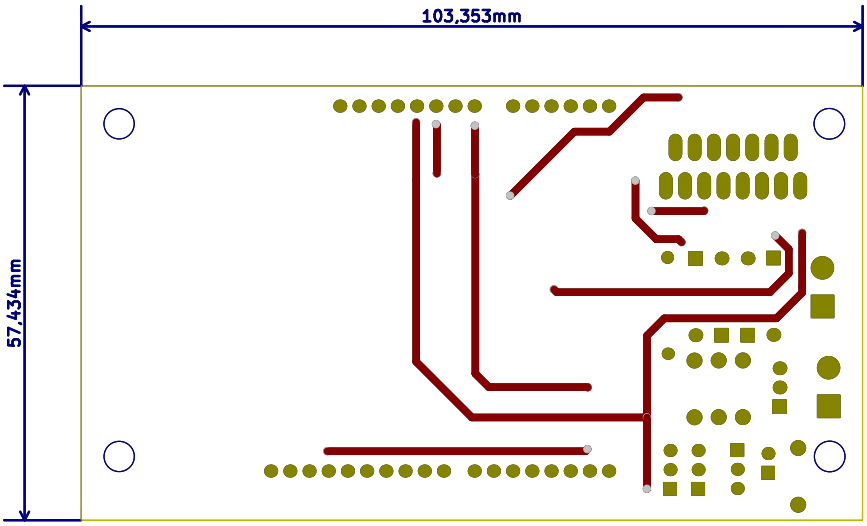


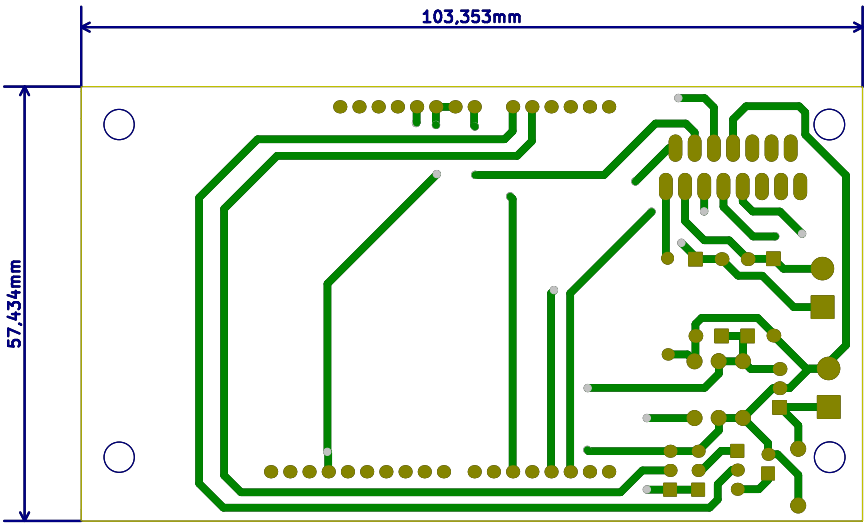


	FECHA	NOMBRE		ESCUELA TÉCNICA SUPERIOR DE INGENIERÍA INDUSTRIAL *Universidad de La Rioja* Grado en Ingeniería Electrónica Industrial y Automática	
Dibujado	04/07/2018	Miguel de Gregorio Santamarina			
Comprobado					
Dib. S. Norma	U.N.E.	Tolerancia general			

Escalas:	**TRABAJO FIN DE GRADO**	**NÚMERO: 6**
PROYECCIÓN	**PRIMERA PCB IMPLEMENTADA EN EL CIRCUITO**	

CREADO CON UNA VERSION PARA ESTUDIANTES DE AUTODESK





	FECHA	NOMBRE		ESCUELA TÉCNICA SUPERIOR DE INGENIERÍA INDUSTRIAL *Universidad de La Rioja* Grado en Ingeniería Electrónica Industrial y Automática	
Dibujado	04/07/2018	Miguel de Gregorio Santamarina			
Comprobado					
Dib. S. Norma	U.N.E.	Tolerancia general			

Escalas:	TRABAJO FIN DE GRADO	NÚMERO: 7
PROYECCIÓN	SEGUNDA PCB IMPLEMENTADA EN EL CIRCUITO	

CREADO CON UNA VERSION PARA ESTUDIANTES DE AUTODESK

"DISEÑO, DESARROLLO Y CONSTRUCCIÓN DE SISTEMA DE CONTROL PARA SERVOVÁLVULA"

DOCUMENTO Nº 5 PLIEGO DE CONDICIONES

Peticionario:	Universidad de La Rioja
Informantes:	Miguel de Gregorio Santamarina
	Estudiante de Ingeniería Electrónica Industrial y Automática

ÍNDICE

PLIEGO DE CONDICIONES

1. Introducción al pliego de condiciones

El autor de este proyecto ha cursado los estudios de Grado en Ingeniería Electrónica Industrial y Automática en la Universidad de La Rioja, cumpliendo con la normativa establecida por la Escuela Superior de Ingeniería Industrial en la normativa de trabajo fin de grado.

El objeto de este Pliego de Condiciones es recoger y establecer todas las disposiciones técnicas, administrativas y económicas, y las normativas que ha de regir este proyecto.

El diseño de este proyecto y sus características han sido descritos en detalle en la memoria del proyecto y sus anexos.

Las condiciones que se especifican en este documento tratan de cumplir con la calidad esperada para este proyecto. En caso de no realizarse según estas condiciones, el proyectista no se responsabilizará de los fallos o averías que puedan ocasionarse en su funcionamiento, y los problemas derivados repercutirían sobre terceras personas.

Todas las modificaciones que puedan sufrir el proyecto y sus documentos deberán ser aprobadas por el ingeniero o proyectista.

2. Condiciones generales

Este proyecto se ajusta a los reglamentos y normativas electrónicas vigentes. Una vez terminado el proyecto se podrán llevar a cabo modificaciones, pero siempre bajo la supervisión del proyectista.

La propiedad intelectual del autor y director del Trabajo Fin de Grado se regirá por el Real Decreto Legislativo 1/1996, de 12 de abril, por el que se aprueba el texto refundido de la Ley de Propiedad Intelectual.

3. Condiciones administrativas

El proyecto constará de los siguientes documentos:

- Un Índice General que indicará la página de comienzo de cada uno de los documentos que forman el proyecto.
- Una Memoria donde se considerarán las necesidades a satisfacer y los factores técnicos a tener en cuenta entrando en profundidad en las posibles soluciones técnicas y en la justificación de la solución elegida.
- Anexos donde se recogerá la documentación considerada de interés para ampliar la descripción detallada de los componentes del sistema.

- Pliego de Condiciones, este documento en el que se establecen las diferentes condiciones técnicas, económicas y administrativas para que proyecto pueda materializarse, evitando posibles malinterpretaciones.
- Presupuesto donde se recogerá el coste de todos los componentes utilizados y la suma total que, junto a la mano de obra, dará el coste final del proyecto. Dicho presupuesto contiene la valoración económica global, desglosada y ordenada por partidas.

4. Normativa y Reglamentación

El proyecto estará regido tanto por la normativa española como por la internacional:

4.1. Reglamento relacionado a productos electrónicos

Respecto al desarrollo de productos electrónicos, se pueden encontrar en AENOR (Asociación Española de Normalización y Certificación) las siguientes normativas:

- Norma UNE1302—2:1973 Vocabulario electrotécnico. Electrónica.
- Norma UNE-EN611000-4-3-1998. Compatibilidad electromagnética.

Este proyecto debido a sus características se rige bajo el reglamento de Baja Tensión:

"Se calificará como instalación eléctrica de baja tensión todo conjunto de aparatos y de circuitos asociados en previsión de un fin particular: producción, conversión, transformación, transmisión, distribución o utilización de la energía eléctrica, cuyas tensiones nominales sean iguales o inferiores a 1000 V para corriente alterna y 1500 V para corriente continua."

4.2. Normativa relacionada con materiales y equipos

Los materiales y equipos de este proyecto deben cumplir los estándares nacionales e internacionales en vigor y obligado cumplimiento. Entre otros:

- UNE 20-324 Grados de protección de los envolventes del material eléctrico de baja tensión.
- UNE 20-334 Conductos para instalaciones eléctricas.
- UNE 21-401 Conductores eléctricos aislados.
- UNE 21-402 Conductores eléctricos aislados y desnudos.

5. Condiciones facultativas

5.1. Dirección

La dirección del montaje de la estructura será llevada a cabo, en su totalidad por el ingeniero proyectista, o por cualquier otra persona en la que éste delegue, atendiendo a la capacidad de dicha persona para responsabilizarse de dicha dirección.

Una vez realizada la instalación, ésta podrá ser utilizada por cualquier persona con conocimientos demostrables suficientes sobre el sistema y sus componentes, las tecnologías implicadas y el funcionamiento global y, por partes, del sistema.

En caso de avería o pérdida de información por una utilización incorrecta, el ingeniero proyectista o la persona en la que haya delegado la dirección del proyecto, quedan exentas de responsabilidad.

5.2. Libro de órdenes

El montaje e instalación de todos los elementos que componen el proyecto se realizará atendiendo al siguiente orden de prioridad en caso de que haya alguna contradicción:

- Presupuesto.
- Pliego de condiciones
- Memoria

Este libro de órdenes debe estar conforme al Decreto 462/1971 de 11 de marzo, y la Orden de 9 de junio de 1971.

5.3. Modificaciones

Si fuera necesario realizar alguna modificación en el presente proyecto, deberá comunicarse con anterioridad a su realización al Ingeniero Director, quién deberá dar la correspondiente autorización.

En caso de realizarse modificaciones en la instalación que no hayan sido previamente comunicadas y autorizadas por el ingeniero Director, las consecuencias que dichos cambios puedan ocasionar serán de total responsabilidad del instalador que las realice.

Respecto a los cambios en la instalación realizados por el propietario de la misma, no serán tratados de forma especial y, en ningún caso, quedan eximidos de la autorización del ingeniero Director.

6. Condiciones de materiales y equipos

A continuación, se detallan las condiciones tanto de hardware como de software con las que hay que contar para hacer uso de la aplicación del proyecto.

6.1. Condiciones técnicas de los materiales

Todos los materiales y componentes, utilizados en el proyecto, deben cumplir todas las especificaciones técnicas que aparecen descritas en la Memoria y deben cumplir asimismo todas las normas descritas en este pliego de condiciones.

Si se considera necesario reemplazar algún componente o material por otro, los nuevos deberán tener las mismas características que los reemplazados, inhibiéndose el ingeniero proyectista de cualquier responsabilidad por fallo, si no se cumplen estos requisitos.

Para el desarrollo e implementación del proyecto se deberá disponer de un PC que será el encargado de recoger los datos, analizarlos y procesarlos. Para ello deberá disponer al menos del siguiente software:

- Windows 10
- Matlab – Simulink R2014b
- Kicad 4.0.7.

7. Condiciones económicas

7.1. Errores en el proyecto

En el caso de existir algún tipo de error en el proyecto se avisará inmediatamente al proyectista y se le informará con detalle de los errores encontrados.

Además, se dejará de usar la aplicación hasta que los errores queden solucionados para prevenir cualquier tipo de daño.

7.2. Liquidación

Terminada la instalación del sistema de la servoválvula, se procederá a la liquidación final, en la que se incluye el importe de las unidades de realización, así como las posibles modificaciones del proyecto que hayan sido aprobadas por la dirección del proyecto.

Al suscribir el contrato, el cliente habrá de abonar el 80% del presupuesto. El 20% quedará como garantía durante los seis primeros meses, a partir de la fecha de puesta en marcha del sistema.

Si transcurridos seis meses desde la puesta en marcha no se ha manifestado ningún defecto o error de funcionamiento, el cliente abonará el 20% que estaba pendiente. A partir de ese momento, se considerarán concluidos los compromisos entre ambas partes, a excepción del periodo de garantía.

8. Disposición final

Las partes contratantes, tanto la dirección del proyecto como la empresa cliente, se ratifican en el contenido del presente pliego de condiciones, que tiene igual validez, a todos los efectos, que una escritura pública, prometiendo su fiel cumplimiento.

"DISEÑO, DESARROLLO Y CONSTRUCCIÓN DE SISTEMA DE CONTROL PARA SERVOVÁLVULA"

DOCUMENTO Nº 6 MEDICIONES

Peticionario:	Universidad de La Rioja
Informantes:	Miguel de Gregorio Santamarina
	Estudiante de Ingeniería Electrónica Industrial y Automática

ÍNDICE

MEDICIONES

C01: Materiales para la construcción de la maqueta		
C01.1	82861010. - Motorreductor DC, Ovoide, Con Escobillas, 3 W, 54 rpm, 0.5 N-m	1
C01.2	Potenciómetro panel multivuelta Bourns - 10k ohmios - 3590S-2-103	1
C01.3	Estructura servoválvula	1
C01.4	Válvula de aguja	1
C01.5	Kit Arduino UNO	1
C01.6	Driver L298N	3
C01.7	Arduino UNO	1
C01.8	Fuente de alimentación 12-40 V	1
C01.9	Corona circular	1
C01.10	Soporte del encoder	1

C02: Mano de obra		
C02.1	Horas de investigación	180
C02.2	Horas de diseño	120
C02.3	Horas de montaje	70
C02.4	Horas de programación	210

"DISEÑO, DESARROLLO Y CONSTRUCCIÓN DE SISTEMA DE CONTROL PARA SERVOVÁLVULA"

DOCUMENTO Nº 7 PRESUPUESTO

Peticionario:	Universidad de La Rioja
Informantes:	Miguel de Gregorio Santamarina
	Estudiante de Ingeniería Electrónica Industrial y Automática

ÍNDICE

Presupuesto

1. Cuadro de precios

C01: Materiales para la construcción de la maqueta		
C01.1	82861010. - Motorreductor DC, Ovoide, Con Escobillas, 3 W, 54 rpm, 0.5 N-m	63,40 €
C01.2	Potenciómetro panel multivuelta Bourns - 10k ohmios - 3590S-2-103	8,10 €
C01.3	Estructura servoválvula	50,00 €
C01.4	Válvula de aguja	10,00 €
C01.5	Kit Arduino UNO	30,00 €
C01.6	Driver L298N	6,00 €
C01.7	Arduino UNO	10,00 €
C01.8	Fuente de alimentación 12-40 V	55,00 €
C01.9	Corona circular	1,00 €
C01.10	Soporte del encoder	1,00 €

C02: Mano de obra		
C02.1	Horas de investigación	50
C02.2	Horas de diseño	30
C02.3	Horas de montaje	40
C02.4	Horas de programación	55

2. Presupuesto

CÓDIGO	DESCRIPCIÓN	CANTIDAD	PRECIO	IMPORTE
C01: MATERIALES PARA LA CONSTRUCCIÓN DE LA MAQUETA				
C01.1	82861010. - Motorreductor DC, Ovoide, Con Escobillas, 3 W, 54 rpm, 0.5 N-m	1	63,40 €	63,40 €
C01.2	Potenciómetro panel multivuelta Bourns – 10k ohmios - 3590S-2-103	1	8,10 €	8,10 €
C01.3	Estructura servoválvula	1	50,00 €	50,00 €
C01.4	Válvula de aguja	1	10,00 €	10,00 €
C01.5	Kit Arduino UNO	1	30,00 €	30,00 €
C01.6	Driver L298N	3	6,00 €	18,00 €
C01.7	Arduino UNO	1	10,00 €	10,00 €
C01.8	Fuente de alimentación 12-40 V	1	55,00 €	55,00 €
C01.9	Corona circular	1	1,00 €	1,00 €
C01.10	Soporte del encoder	1	1,00 €	1,00 €
C01.11	PCB circuitos	2	10,00 €	20,00 €
			Total capítulo	266,50 €

CÓDIGO	DESCRIPCIÓN	CANTIDAD	PRECIO	IMPORTE
C02: Mano de obra				
C02.1	Horas de investigación	180	40,00 €	7200,00 €
C02.2	Horas de diseño	120	35,00 €	4200,00 €
C02.3	Horas de montaje	70	30,00 €	2100,00 €
C02.4	Horas de programación	210	35,00 €	7350,00 €
			Total capítulo	20850,00 €

3. Resumen del presupuesto

CAPÍTULO RESUMEN		EUROS	%
C01	Materiales para la construcción de la maqueta	266,50 €	1,3
C02	Mano de obra	20850,00 €	98,7

TOTAL EJECUCIÓN MATERIAL	21116.50
13% Gastos generales	2745.145
6% Beneficio industrial	1266,99
SUMA DE GG y BI	4012,135
21% de IVA	5277,013
TOTAL PRESUPUESTO GENERAL	30405,65

Asciende el presupuesto general a la expresada cantidad de TREINTA MIL CUATROCIENTOS CINCO EUROS CON SESENTA Y CINCO CÉNTIMOS.

FIRMADO:

D. Miguel de Gregorio Santamarina

Logroño a 6 de julio de 2018